Departamento de Ingeniería Técnica de
Telecomunicación Telemática

PROYECTO FIN DE CARRERA

SISTEMA DE GESTIÓN REMOTA DE TÍTULOS DE TRANSPORTE

Autor: Patricia Mozas Vozmediano

Tutor: Raúl Sánchez Reillo

Leganés, Octubre de 2015

Título: SISTEMA DE GESTIÓN REMOTA DE TÍTULOS DE TRANSPORTE
Autor: Patricia Mozas Vozmediano
Director: Raúl Sánchez Reillo

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día __ de _____ de 20__ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

A mis padres, Julio y Rosi, quienes me animaron a comenzar esta carrera y a no rendirme a mitad de camino. A mi hermana, Natalia, por su espíritu positivo y por motivarme a superarme con su ejemplo.

A mi tutor en Gemalto, David, por su colaboración en este proyecto. Gracias por enseñarme a diario y transmitir tanto entusiasmo por la profesión de Ingeniero, el mundo de las tarjetas y la seguridad.

A mi tutor y profesor, Raúl, por perseguirme en el último momento hasta otro país y no dejarme tirar la toalla.

Y en especial, a Javier, por sus conocimientos, apoyo incondicional y paciencia. Sin él este proyecto no sería posible.

Resumen

Este proyecto describe la arquitectura de un sistema de gestión remota de títulos de transporte basado en el ecosistema NFC y en estándares de la industria como MIFARE4Mobile y GlobalPlatform. El sistema combina una aplicación web, que permite al usuario lanzar peticiones de compra, recarga o borrado de tarjetas de transporte MIFARE, con una aplicación servidor que recibe las peticiones y las formatea añadiendo la seguridad necesaria para descargarlas en un móvil NFC.

Abstract

This project describes the architecture of a remote management system for transport ticketing based on NFC mobile ecosystems and standards as MIFARE4Mobile and Global Platform. The system combines a web application that allows the user to request the purchase, refill or deletion of MIFARE cards, with a server application that receives and process the requests by adding the security and confidentiality required to download them on a mobile NFC.

:

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción y objetivos

1.1 Motivación del proyecto

La entidad para la que trabajo, Gemalto, ofrece soluciones NFC líderes en el mercado para pagos y servicios móviles sin contacto, que permiten a los operadores de redes móviles, bancos, redes de transporte, pequeño comercio y otros proveedores de servicios brindar atractivas experiencias NFC de una manera segura.

Concretamente en España, trabajando como consultor técnico especializado en tarjeta SIM, he tenido la oportunidad de presenciar el despliegue comercial del uso del teléfono móvil como medio de pago en un caso de uso concreto: el transporte público.

Tras varias experiencias piloto a nivel nacional, Valencia lanza en Julio del 2014 el billete electrónico en la SIM que da acceso tanto a la red de autobuses de la Empresa Municipal de Transportes (EMT) como a la de ferrocarriles de la Generalitat Valenciana (FGV). Le sigue Logroño en Diciembre de 2014 y en Marzo de 2015 Telefónica anuncia un proyecto en conjunto con el Consorcio Regional de Transportes de Madrid (CRTM) con el mismo fin para nuestra capital.

Sin embargo la tarjeta SIM, mi ámbito de experiencia, es sólo un eslabón en la gran cadena que pone en funcionamiento la gestión de billetes de transporte en el móvil. ¿Cuál es el ecosistema que hay detrás? ¿Quiénes son los actores? ¿Cómo viaja la información a través de las redes de comunicación cuándo se compra o recarga un billete electrónico? ¿Qué seguridad se aplica? ¿Qué tecnologías sin contacto y estándares se utilizan hoy en España? Para responder a todas estas preguntas decido ponerme en la piel de un proveedor de servicios de transporte al que llamo "Nubetrans" y diseñar e implementar un sistema que gestione los billetes electrónicos de diferentes operadores de transporte a nivel nacional a través de una aplicación web. Una vez registrado, el usuario podrá:

- Obtener y recargar sus títulos de transporte desde la web sin necesidad de acudir a un punto de venta o recarga.
- Pagar a bordo del servicio de transporte público con su teléfono móvil exactamente de la misma manera que con su tarjeta sin contacto.

1.2 Objetivos

El objetivo principal de este proyecto es investigar sobre la tecnología móvil NFC como medio de acceso a la red de transportes mediante el desarrollo de un prototipo final que permita la gestión remota de billetes en el móvil respetando el ecosistema y las normas de seguridad definidas por GlobalPlatform. La implementación de dicho prototipo debe adaptarse a las infraestructuras de transporte basadas en MIFARE y su funcionalidad debe ser probada en tarjetas SIM y teléfonos reales. Con esta solución se quiere demostrar las posibilidades reales de migración de los sistemas actuales de compra y canjeo, dependientes de puntos de venta y de tarjetas físicas, a un ecosistema móvil.

A continuación se listan los objetivos concretos a cumplir para este proyecto:

1. Comprobar que a través de una aplicación web cualquier persona desde un PC con conexión a internet puede comprar y gestionar en su móvil y tarjeta SIM NFC los títulos que le darán acceso a una red de transporte.

2. Comprobar que se puede implementar un TSM (Trusted Service Manager) con los comandos de gestión remota de MIFARE4Mobile v1.01 **3** y con seguridad SCP02 de GlobalPlatform **1** que acceda vía proxy HTTP(S) en el móvil a una tarjeta SIM NFC y que actualice su memoria Mifare.

3. Validar la personalización realizada en la tarjeta NFC mediante la ejecución de una aplicación en el PC que permitirá leer el mapa de memoria Mifare y con otra que simulará el acceso a la red de transportes. Para comprobar que la carga del título de transporte se hizo correctamente, insertaremos la tarjeta en un móvil NFC que acercaremos a una lectora sin contactos conectada a dicha aplicaciones en el PC.

4. Determinar la infraestructura necesaria de acuerdo con GlobalPlatform para gestionar una aplicación de transporte en una tarjeta inteligente satisfaciendo los requerimientos de negocio y seguridad de todas las entidades implicadas (operadores móviles y operadores de transporte).

5. Documentar las conclusiones obtenidas sobre las limitaciones encontradas a la hora de implantar un sistema de personalización de aplicaciones de terceros instaladas en el elemento central de un operador móvil, la tarjeta SIM.

La finalidad de esta memoria es que todos los aspectos que se han plasmado en estos objetivos queden correctamente descritos en las distintas secciones que la componen. En la sección siguiente se detalla cómo se han estructurado los distintos capítulos para ello.

1.3 Estructura de la memoria

Para facilitar la lectura de la memoria, se incluye a continuación un breve resumen de cada capítulo:

- El Capítulo 2 presenta el estado del arte de las tecnologías que definen el marco en el que nace la idea del proyecto. Se consideran conceptos como Tarjetas Inteligentes, NFC y GlobalPlatform, organización que se preocupa por tener una infraestructura estandarizada para el desarrollo, despliegue y gestión de tarjetas inteligentes y que nos permite entender todos los actores de un ecosistema NFC y las relaciones entre ellos.

- El Capítulo 3 constituye una descripción de las tecnologías necesarias para el desarrollo completo del proyecto. Entre estos conceptos se incluyen: PHP como lenguaje de programación para páginas web dinámicas, SQL como lenguaje de gestión de bases de datos, Spring como framework de aplicaciones Java,, REST para describir la interfaz entre sistemas que utilizan HTTP para obtener datos o

tecnologías más específicas como MIFARE4Mobile que permite la gestión remota de tarjetas MIFARE en un entorno móvil.

- En el Capítulo 4 y en el Capítulo 5 se desarrolla el prototipo principal objeto de este proyecto, Nubetrans.

- En el Capítulo 6 se describe la historia del proyecto formada por la distribución temporal del mismo y por la elaboración de un presupuesto.

- En el Capítulo 7 se resumen las principales conclusiones del proyecto realizado y se enumeran una serie de posibles líneas de trabajo futuro.

Finalmente se incluyen las referencias consultadas en el desarrollo del proyecto, que incluyen las especificaciones a las que ha sido necesario recurrir para implementar aspectos concretos de las aplicaciones desarrolladas, libros y recursos web.

Capítulo 2

Estado del Arte

2.1 Introducción

En este capítulo se procede a explicar las tecnologías involucradas para que el lector comprenda el marco en el que se desarrolla el proyecto.

Comenzaremos con una breve introducción a las tarjetas inteligentes, que permiten al usuario llevar consigo una seguridad portable para realizar pagos, como en el caso de las tarjetas bancarias, ejecutar aplicaciones de forma segura, como las tarjetas Java Card y que incluso son capaces de jugar el papel de token de autenticación, como en el caso de las tarjetas SIM en las redes de telefonía móvil.

A continuación presentaremos la tecnología NFC (Near Field Communication), que permite la comunicación inalámbrica a corta distancia (de 0 a 20 cm, generalmente 10 cm) de dispositivos que operan a una frecuencia de 13.56 MHz. Dado que su tasa de transferencia es de **424 kbit/s,** más que para la transmisión de grandes cantidades de datos, está enfocada a la **comunicación instantánea**, es decir, identificación y validación de equipos/personas.

Por último hablaremos de GlobalPlatform, una organización independiente, sin ánimo de lucro, que fija estándares a nivel mundial en infraestructuras que incorporan el uso de tarjetas inteligentes (smart cards). GlobalPlatform busca asentar las bases que permitan a todas las entidades del mercado implicadas, ya sean operadores móviles, operadores de transporte o bancos, gestionar múltiples aplicaciones dentro del entorno de trabajo de estos dispositivos.

2.2 Tarjetas Inteligentes

Una tarjeta inteligente (smart card), o tarjeta con circuito integrado (TCI), es cualquier tarjeta del tamaño de un bolsillo con circuitos integrados que permiten la ejecución de cierta lógica programada. Aunque existe un diverso rango de aplicaciones, hay dos categorías principales de TCI:

- **Tarjetas de Memoria**: son aquellas que contienen sólo componentes de memoria no volátil y posiblemente alguna lógica de seguridad.

- **Tarjetas Inteligentes**: son aquellas que están basadas en un microprocesador y una memoria. La percepción estándar de una tarjeta inteligente es una tarjeta con microprocesador de las dimensiones de una tarjeta de crédito (o más pequeña, como por ejemplo, tarjetas SIM) con varias propiedades especiales (ej. un procesador criptográfico seguro, sistema de archivos seguro, características legibles por humanos) y es capaz de proveer servicios de seguridad (ej. confidencialidad de la información en la memoria).

En ambas categorías, se puede establecer una nueva clasificación dependiendo del método de comunicación empleado:

- **Con Contactos**: la comunicación se hace desde la conexión física de cada uno de los contactos metálicos de la tarjeta con los correspondientes de la lectora. Los contactos metálicos están debidamente estandarizados en la parte 2 de la norma ISO/IEC 7816).

Figura 1 Contactos ISO/IEC 7816-2 de un tarjeta con contactos [10]

- **Sin Contactos**: la comunicación se realiza de forma aérea mediante radiofrecuencia. Por tanto, no es necesario un contacto físico entre la tarjeta y la lectora. A las tarjetas de memoria sin contactos se las suele denominar Tags, y sirven fundamentalmente para la identificación, ya sea de animales, personas u cosas. En caso de que se trate de Tarjetas Inteligentes, se las denomina Tarjetas sin Contactos y son muy interesantes para el control de acceso a edificios y/o transporte público. El estándar de comunicación de tarjetas inteligentes sin contacto es el ISO/IEC 14443 del 2001.

- **Tarjetas combi con/sin contactos**: son una mezcla de los dos tipos anteriores, en los que se mezclan las ventajas de seguridad de una tarjeta con contactos, realizándose dicha comunicación mediante el mismo tipo de conexiones, con las ventajas de velocidad de las tarjetas sin contactos, comunicándose entonces mediante una antena. Estas tarjetas contemplan sólo la inclusión de Tarjetas Inteligentes (es decir, microprocesador), y su gran potencial de aplicación radica en sistemas de acceso y de transporte público, en el que participan entidades financieras u otro tipo de instituciones, que prefieran que el trato de sus datos se haga de forma segura.

Y si hacemos referencia a la estructura de su sistema operativo:

- **Nativas**: Basadas en sistemas de ficheros, aplicaciones y comandos. Estas tarjetas disponen del equivalente a un sistema de ficheros compatible con el estándar ISO/IEC 7816 parte 4 y un sistema operativo en el que se incrustan una o más aplicaciones (durante el proceso de fabricación) que exponen una serie de comandos que se pueden invocar a través de APIs de programación

- **JavaCard**: La tecnología Java Card combina parte del lenguaje de programación Java con un entorno de ejecución optimizado para Smart Cards y similares. El objetivo de la tecnología Java Card es llevar los beneficios del desarrollo de software en Java al mundo de las Smart Cards. Java Card es compatible con otros estándares desarrollados para tarjetas inteligentes como el ISO 7816[18].

A continuación se profundiza en dos tipos de tarjetas claves en el ámbito de transporte, las tarjetas sin contacto de la familia MIFARE de Philips que se utilizan actualmente como medio de acceso en un gran número de redes de transporte público y las tarjetas inteligentes SIM, que trasportan los credenciales para conectarse a las redes de telefonía móvil y que ofrecen un medio alternativo para utilizar MIFARE en el teléfono móvil.

2.2.1 Tarjetas MIFARE

Figura 2 Tarjeta Mifare. 12

MIFARE es la tecnología de tarjetas inteligentes sin contacto (TISC) más ampliamente instalada en el mundo con aproximadamente 250 millones de TISC y 1.5 millones de módulos lectores vendidos.

Es equivalente a las 3 primeras partes de la norma ISO 14443 Tipo A de 13.56 MHz con protocolo de alto nivel. La distancia típica de lectura es de 10 cm (unas 4 pulgadas). La distancia de lectura depende de la potencia del módulo lector, existiendo lectores de mayor y menor alcance.

La tecnología MIFARE es propiedad de NXP Semiconductores (antes parte de Philips Semiconductores). La tecnología es económica y rápida, razón por la cual es la más usada a nivel mundial.

Las tarjetas MIFARE son tarjetas de memoria protegida. Están divididas en sectores y bloques y mecanismos simples de seguridad para el control de acceso. Su capacidad de cómputo no permite realizar operaciones criptográficas o de autenticación mutua de alto nivel, a excepción de MIFARE Desfire como se explica más adelante, estando principalmente destinadas a monederos electrónicos simples, control de acceso, tarjetas de identidad corporativas, tarjetas de transporte urbano o para ticketing.

Cada sector se divide en cuatro bloques, de los cuales tres pueden contener información del usuario. La información es de formato libre, y se puede modificar con comandos simples de lectura y escritura. Mifare provee un formato especial llamado 'bloque de valor'; los bloques que tienen información guardada en este formato se comportan de una forma diferente, incluyendo operaciones de descuento e incremento.

Los sectores utilizan dos claves de acceso llamadas 'A' y 'B'. Estas llaves se almacenan en el cuarto bloque junto con los permisos de acceso a cada uno de los tres bloques. Estos permisos pueden ser: lectura, escritura, descuento o incremento (para bloques de valor).

Cada tarjeta contiene un número único de identificación permanente (UID) grabado en el bloque 0 del sector 0 normalmente durante el proceso de fabricación. El UID no está cifrado y cualquier lector que sea compatible con la norma ISO puede leer el número. La norma ISO 14443-3ª define 3 tipos de UID: de 4, 7 y 11 bytes,

Después de establecer un canal cifrado la tarjeta envía un código de identificación de conexión, que usualmente es el número de serie de la tarjeta aunque la norma ISO 14443 dice que este número puede ser aleatorio. Con este número de conexión el lector está en capacidad de realizar cualquier operación en la tarjeta, previa presentación de las claves de acceso a los respectivos sectores.

A continuación se muestra a modo de ejemplo el mapa de memoria de una tarjeta MIFARE Classic 1KB:

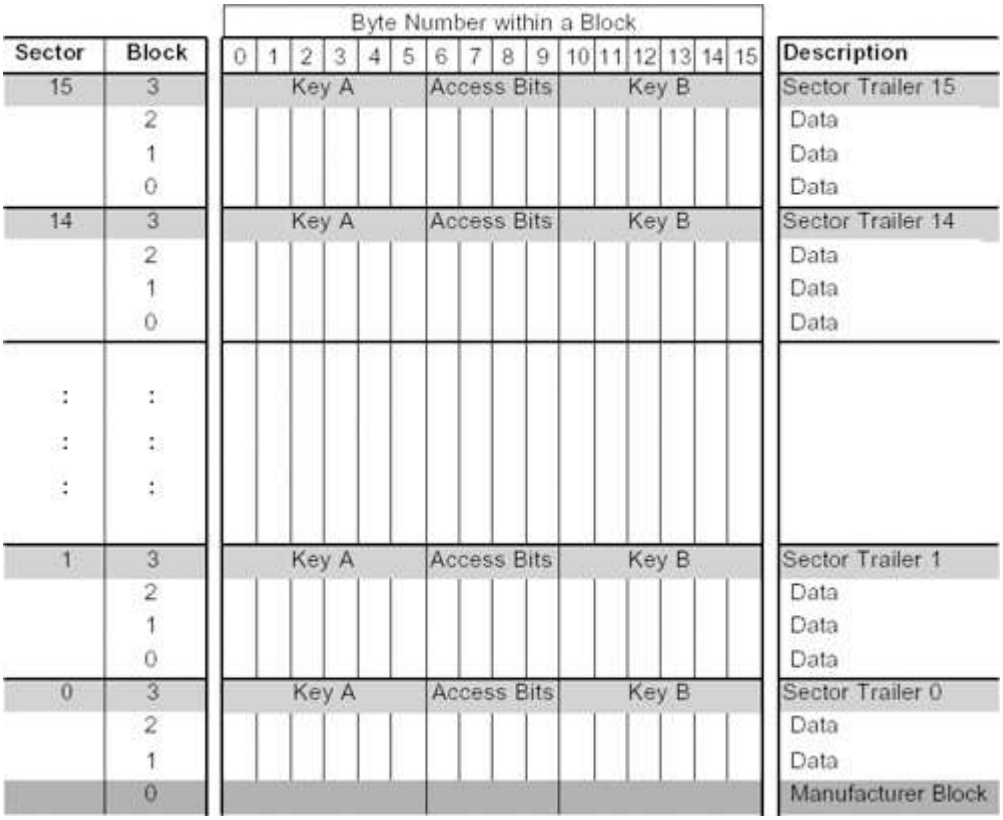


Figura 3 Mapa de memoria de una tarjeta MIFARE Classic 1KB. [11]

Variantes:

- **MIFARE Classic**. Son fundamentalmente dispositivos de almacenamiento de memoria. Existen tarjetas de 1KB y de 4KB. La MIFARE Standard de 1KB ofrece unos 768 bytes de almacenamiento de datos, dividida en 16 sectores. La MIFARE Standard de 4k ofrece 3 KB dividido en 64 sectores.

- **MIFARE Ultralight**. Es semejante a la Classic, pero sólo tiene 512 bits de memoria (es decir 64 octetos), sin seguridad. Esta tarjeta es muy barata así que se utiliza a menudo de forma desechable.

- **MIFARE T = CL.** Bajo esta denominación se encuadran las tarjetas Mifare ProX y SmartMX. Son tarjetas microprocesadas que incorporan un sistema operativo de tarjeta (Card Operating System - COS) y aplicaciones desarrolladas específicamente para ser ejecutadas en la tarjeta. Estas tarjetas son capaces de ejecutar operaciones complejas de forma rápida y segura, igual que las tarjetas con contactos ISO 7816.

- **MIFARE DESFire**. Esta tarjeta es una versión especial de Philips SmartMX. Se vende con un software de propósito general incorporado (el sistema operativo DESFire), que ofrece más o menos las mismas funciones que Mifare Standard (4kB de almacenamiento de datos dividido en 16 bloques), pero con una mayor flexibilidad, una mayor seguridad (triple DES), y con mayor rapidez (protocolo T=CL).

- **MIFARE DESFire EV1**. Es la primera evolución de MIFARE DESFire, compatible con la versión anterior, pero aún más segura ya que utiliza AES, alcanzando la certificación EAL 4 +.

Algunos ejemplos de sistemas MIFARE:

- Metro de Moscú: Primer Gran Sistema de Transporte del Mundo en operar totalmente tarjetas sin contacto con tecnología MIFARE, con 9 millones de personas al día.
- Tarjeta Apunt en el metro de valencia y demás operadores de transporte del área metropolitana de Valencia (desde 2009).
- Tarjeta Apunt en el tranvía de alicante y demás medios de transporte público del área metropolitana de Alicante (desde 2007).
- Tarjeta Oyster en Londres con más de 10 millones de tarjetas en circulación en el 2007.
- Tarjeta sube-T (DesFire) del Consorcio Regional de Transportes de Madrid (CRTM).

2.2.2 La tarjeta SIM

Una **tarjeta SIM** (acrónimo en inglés de Subscriber Identity Module, en castellano módulo de identificación del suscriptor) es una tarjeta inteligente con contactos usada en teléfonos móviles y módems HSDPA (High-Speed Packet Access) o LTE que se conectan al puerto USB.

El uso de la tarjeta SIM es obligatorio en las redes GSM. Su equivalente en las redes UMTS se denomina USIM o UICC (acrónimo de Universal Integrated Circuit Card, 'Tarjeta Universal de Circuito Integrado'), siendo también popular el RUIM (Removable User Identify Module, 'Módulo de Identidad de Usuario Desmontable') en los teléfonos CDMA.

Las tarjetas SIM / USIM proporcionan diversos servicios al ecosistema de la Telefonía móvil:

- **Desde el punto de vista de seguridad**, almacenan las claves y realizan los cálculos criptográficos necesarios para gestionar la autenticación del usuario final. También proporcionan un sistema de códigos y claves administrativas para administrar diferentes niveles de acceso, incluyendo el PIN para el usuario final. La gestión remota de la tarjeta, tanto de contenido de ficheros como de descarga de aplicaciones, también es posible gracias a las capacidades criptográficas de la tarjeta y la facilidad para almacenar claves para esta función.

- **Desde el punto de vista del operador**, proporcionan un sistema de almacenamiento que configura ciertas funciones del terminal móvil, como el número del centro de mensajes cortos, las redes preferidas a las que el móvil debe conectarse en caso de no estar la red del operador presente, APN, el IMSI, tiempo entre reconexiones, etc.

- **Desde el punto de vista del usuario**, permiten almacenar la agenda telefónica, los mensajes cortos, los lugares más habituales de conexión para permitir una reconexión rápida en caso de apagado del terminal, etc.

Desde el punto de vista de comunicación, durante mucho tiempo sólo se utilizaron los cinco contactos metálicos descritos al inicio de la sección 2.2 pero la aplicación de nuevas interfaces, implica el uso de los contactos C4 y C8 para conexiones USB, así como un cambio en el uso del C6, clave en la tarjeta SIM NFC. El contacto C6 implementa el protocolo SWP (Single Wire Protocol - TS 102 613) que se explica más adelante en la sección 2.3.2 en relación al móvil NFC.

2.2.2.1 Gestión Remota de la tarjeta SIM

Actualmente la gestión remota de las aplicaciones en la tarjeta SIM, conocida también como gestión OTA (Over The Air), se realiza principalmente a través de SMS, pero también es posible utilizar HTTPS a través de un proxy en el terminal móvil como explicamos a continuación.

En este proyecto se han considerado las siguientes opciones:

- **SMS/BIP/TCP:**
  La comunicación para enviar comandos o un script de personalización a la tarjeta SIM puede ser realizada puramente usando concatenación de SMS (cadena alfanumérica de hasta 140 caracteres) o puede iniciarse con un SMS "push" al terminal móvil para abrir un canal BIP CAT/TP (Bearer Independent Protocol) de mayor ancho de banda. El primer caso implica implementar protocolos de seguridad adicionales como SCP80, y el segundo implementar el servicio BIP CATTP (definido en estándares como TS 102.225 y 102.226 - **28**) lo que complicaría el proyecto en cuestión. Además en ambos casos es necesario contar con una subscripción de red.

- **HTTPS:**
  La comunicación para enviar comandos o un script de personalización a la tarjeta SIM puede ser iniciada con un SMS o un mensaje "push" a un proxy en el móvil que establece una conexión HTTPS con un servidor. El proxy puede enviar los comandos recibidos del servidor a la tarjeta SIM y mandar una respuesta de vuelta al servidor.

2.3 NFC

Near Field Communication (NFC) es una tecnología inalámbrica de corto alcance que permite la comunicación entre dispositivos a una distancia de menos de 10 cm. Evolucionó de las primeras tecnologías sin contactos RFID (identificación por radiofrecuencia) y de las soluciones MIFARE y FeliCa creadas por las empresas Philips y Sony respectivamente. NFC comenzó a consolidarse en 2002, cuando dichas empresas decidieron llegar a un acuerdo para desarrollar un sistema compatible con ambas soluciones que integrara dispositivos electrónicos móviles dando lugar al NFC Forum.

En la comunicación NFC, un dispositivo genera una onda de radio de baja frecuencia que opera a 13,56 MHz, una franja exenta de regulación administrativa; cuando otro dispositivo NFC se acerca lo suficiente para ponerse en contacto, se genera un "acoplamiento magnético inductivo", por medio del cual se puede realizar una

transferencia de energía y de datos entre los dos dispositivos. El acoplamiento inductivo funciona solo para distancias cortas, siendo la principal diferencia entre NFC y otras tecnologías inalámbricas como Bluetooth y WiFi, donde éstas trabajan a una distancia máxima aproximada de 10 metros y 100 metros respectivamente. Debemos aclarar que, aunque parezca lo contrario, NFC es un medio de transmisión inalámbrico seguro: en primer lugar, la reducida distancia de acción hace que sea muy difícil que un tercer dispositivo interfiera en la conexión, por otro lado, las transmisiones pueden usar encriptaciones de seguridad tales como SSL, por lo que los datos viajan a salvo entre dispositivos.

NFC maneja una tasa de transferencia de datos máxima de 424 kbps, por lo que no está pensada para una transmisión masiva de datos como puede darse con las tecnologías Bluetooth (3 Mbps) y WiFi (54M bps). Sin embargo es ideal para el intercambio instantáneo de unos pocos bytes de información, lo justo para que valide e identifique al usuario.

	NFC	RFID	IrDA	Bluetooth
Tiempo de configuración	<0.1ms	<0.1ms	0.5s	6sec
Rango	Hasta 10cm	Hasta 3m	Hasta 5m	Hasta 30m
Usabilidad	Centrada en personas. Fácil, intuitivo, rápido	Centrada en objetos.Fácil	Centrada en datos.Fácil	Centrada en datos. Medio
Selección	alta, dada, seguridad	En parte dada	Campo de visión	¿Quién eres tú?
Casos de Uso	Pagos, acceso, compartir, iniciar servicio	Rastreo de objetos	Control e intercambio de datos	Red para intercambio de datas, auriculares
Experiencia de Usuario	Toque, onda, simple conexión	Obtener información	Fácil	Requiere configuración

Tabla 1 Comparativa NFC con otras tecnologías inalámbricas. 8

Los dispositivos NFC pueden operar en dos modos:

- **Modo Pasivo**: sólo uno de los dispositivos es quien genera el campo magnético, mientras que los otros terminales, absorben la energía para poder comunicarse (caso de las pegatinas).

- **Modo Activo**: cada uno de los dispositivos dispone de una fuente interna de poder y además genera su propio campo magnético para transmitir los datos.

Existen tres modos de funcionamiento NFC y son los siguientes:

- **Modo emulación**: el dispositivo NFC se comporta exactamente como una tarjeta sin contacto y se puede utilizar en sistemas de transporte basadas en MIFARE, Calypso o Felica, así como en los sistemas de pago bancario basados en diferentes métodos como Visa payWave, MasterCard PayPass o American Express ExpressPay

- **Modo lector**: el dispositivo NFC se encuentra en modo activo y lee una etiqueta RFID pasiva, como por ejemplo leer y almacenar una dirección Web o el cupón de un cartel de publicidad interactiva

- **Modo peer to peer (P2P):** dos dispositivos NFC se comunican entre sí para el intercambio de información.

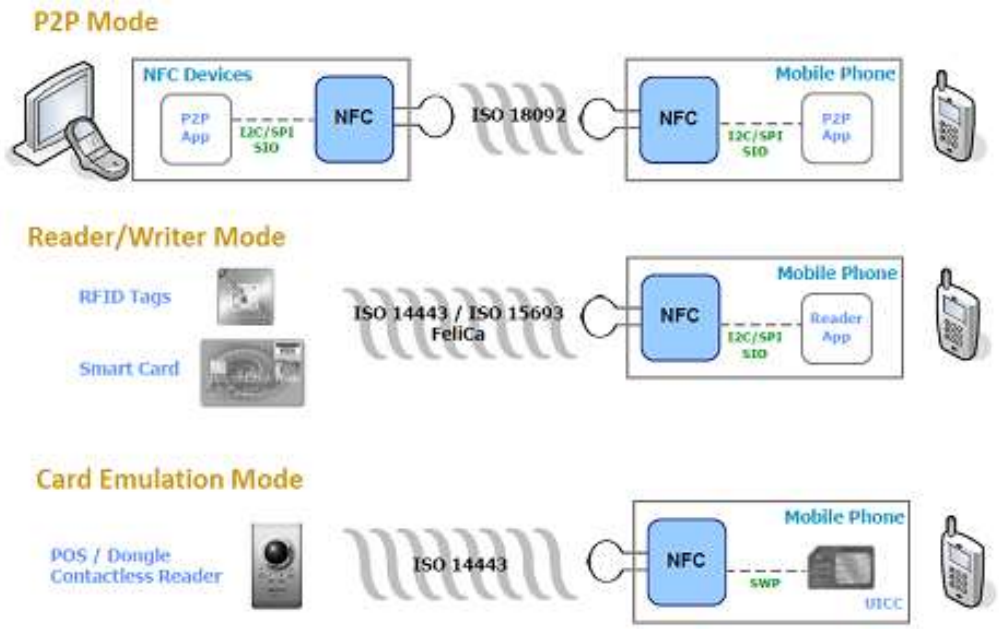


Figura 4 Modos de funcionamiento NFC. 27

Los usos más comunes de esta tecnología son los pagos móviles y los pequeños pagos en el transporte urbano pero existe un rango amplio de posibilidades como el acceso físico a edificios, la identificación de personas en eventos, el check-in en aeropuertos o la acreditación de entrada a conciertos.

Figura 5 Casos de uso de NFC

En definitiva, las siguientes características hacen que NFC sea una tecnología a tener en cuenta:

- **Alcance y disponibilidad**: NFC tiene el potencial para ser integrado en todos los teléfonos móviles del mundo. Mediante dicha integración, multitud de nuevos servicios se ponen al alance cualquier persona a través de su teléfono móvil.

- **Variedad de uso**. NFC se puede usar para varias tareas, desde el pago de bienes, la compra de billetes o emparejamiento de dispositivos para el intercambio de información.

- **Fácil de usar.** NFC sólo requiere que dos dispositivos se toquen con el fin de comunicarse.

- **Seguridad**. las transmisiones pueden usar encriptaciones de seguridad tales como SSL, por lo que los datos viajan a salvo entre dispositivos.

- **Infraestructura**. NFC es compatible con las infraestructuras existentes para tarjetas sin contactos, ya estén basadas en MIFARE/Calypso, como puede ser el caso de plataformas de transporte, o en EMV, como puede ser el caso de lectores de pagos. Dicha compatibilidad facilita el despliegue comercial de NFC, donde además se cuenta con un interfaz de usuario en el teléfono móvil y una conexión a Internet.

2.3.1 Estándares NFC y NFC Forum

Los estándares NFC y de las tarjetas inteligentes son reconocidos por diferentes organismos internacionales como ISO/IEC (International Organization for Standarization (International Electro-technical Commision), ETSI (European Telecomunications Standards Institute) y ECMA (European Association for Standarizing Information and Communication Systems). Estos estándares especifican los esquemas de modulación, codificación, velocidades de transferencia y formato del interfaz RF (Radio-Frecuencia) de los dispositivos NFC. Definen también esquemas de inicialización y condiciones requeridas para control de colisiones de datos durante la inicialización, tanto para modos pasivos como modos activos de NFC, y protocolos de transporte, incluyendo protocolos de activación y métodos de intercambio de datos.

El estándar NFC se registró como ECMA-340 con ECMA Internacional (el organismo de normalización en los campos de la Información y la Comunicación) en diciembre de 2002. En diciembre de 2003, ganó la acreditación internacional ISO / IEC 18092 (NFC IP-1). Más tarde, en enero de 2005, NFC IP-2, el estándar ampliado, se estableció como una acreditación internacional ISO / IEC 21,481.

Como se puede observar en la figura 2.7, NFC se encuentra estandarizado bajo:

- **ISO/IEC 18092 / ECMA-340 / ETSI TS 102 190** Near Field Communication Interface and Protocol-1 (NFCIP-1): Definen la modulación, los esquemas de codificación de bits y la arquitectura para las tasas de transferencia de datos de 106, 212 y 424 kbits/s. Además estandarizan la interfaz de señal de comunicación y el flujo general del protocolo. NFCIP-1 también define los modelos de comunicación, activa y pasiva, como los diferentes tipos de interacción física (modo emulación, modo lector y peer to peer).

- **ISO/IEC 21481 / ECMA-352 / ETSI TS 102 312**: Near Field Communication Interface and Protocol-2 (NFCIP-2): NFCIP-2 provee una puerta de entrada entre diferentes estándares de interfaz existentes. Los dispositivos que implementan NFCIP-2 necesitan implementar funciones de dispositivo de acoplamiento de proximidad (ISO/IEC 14443), dispositivo de acoplamiento de vecindad (ISO/IEC 15693) y las funciones de iniciador y target definidas en ECMA-340. Esto hace que los dispositivos NFC sean compatibles con sistemas existentes de FeliCa, MIFARE y otros.

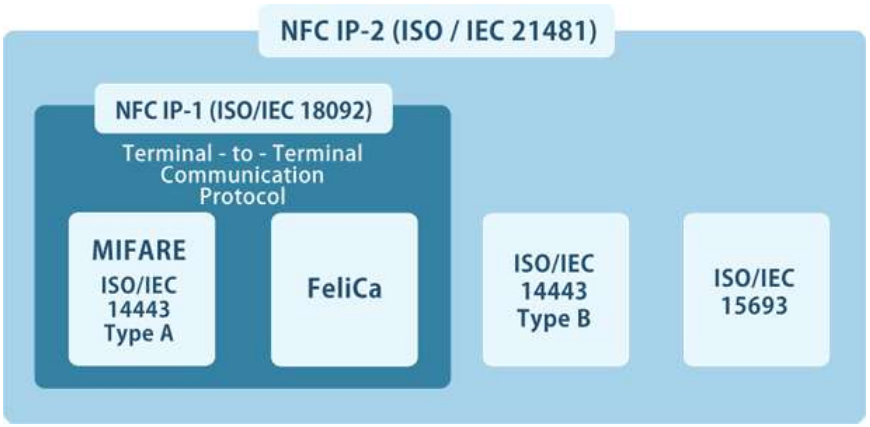


Figura 6 NFC IP-2 Normas ISO/IEC 21481. 24

El **NFC Forum** 5 es una asociación industrial sin ánimo de lucro fundada por NXP Semiconductors (Philips), Sony Corporation y Nokia para regular el uso de la interacción inalámbrica de corto alcance en la electrónica de consumo, dispositivos móviles y los PCs. En junio de 2006, sólo 18 meses después de su fundación, describió formalmente la arquitectura de la tecnología NFC. Hasta septiembre de 2009, ha publicado 12 especificaciones. El NFC Forum promueve la implantación y la estandarización de la tecnología NFC como mecanismo para la interoperabilidad entre dispositivos y servicios. Para conseguir esto, se encarga de:

- Desarrollar especificaciones basadas en estándares.
- Asegurarse del uso de las especificaciones del NFC Forum.
- Trabajar para que los productos con tecnología NFC cumplan con las especificaciones del NFC Forum.
- Educar a los consumidores y las empresas respecto de la tecnología NFC.

El NFC Forum ha establecido un estándar en la que se registra un formato común para poder compartir datos entre los dispositivos NFC o entre dispositivos y etiquetas NFC. Son los siguientes:

- **NFC Data Exchange Format (NDEF)**: Especifica un formato común y compacto para el intercambio de datos. Un mensaje NDEF (véase en la Figura 7) se compone de uno o más registros NDEF. El primer registro de un mensaje es marcado con el flag MB (Message Begin), y el último con el ME (Message End). La longitud mínima de mensaje es un registro, que se consigue estableciendo ambos flags, MB y ME en el mismo registro. El número máximo de NDEF Records que se pueden transportar en un mensaje NDEF es ilimitado.

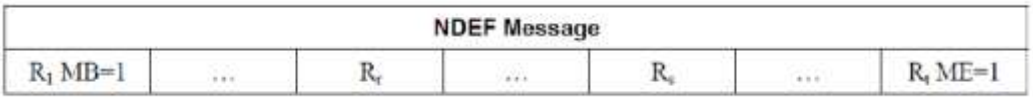


Figura 7 Mensaje NDEF. 7

- **NFC Record Type Definition (RTD)**: Especifica tipos de registros estándar que pueden ser enviados en los mensajes intercambiados entre los dispositivos NFC.

    - Smart Poster RTD: Para posters que incorporen etiquetas con datos (URLs, SMSs o números de teléfono)
    - Text RTD: Para registros que sólo contienen texto
    - Uniform Resource Identifer (URI) RTD: Para registros que se refieren a un recurso de internet.

2.3.2 Móvil NFC

Un teléfono móvil con tecnología NFC está típicamente compuesto de varios circuitos integrados, uno o más elementos seguros (SE) y una interfaz NFC. El sistema se compone de los siguientes elementos:

- CLF (Contactless Front-End) compuesto por el controlador NFC y la antena RFID (compatible con el estándar ISO 14443).
- Controlador Host
- Elemento Seguro (acostumbra a ser una tarjeta SIM)

El elemento seguro puede ser controlado y accedido internamente desde el controlador host o bien desde el campo RF externo. El controlador host es el corazón de cualquier Smartphone. El HCI (Host Controller Interface) crea un puente entre el controlador NFC y el controlador Host. El controlador Host fija la modalidad operativa del controlador NFC a través del HCI, procesa los datos enviados y recibidos y establece la conexión entre el controlador NFC y el SE (elemento seguro).

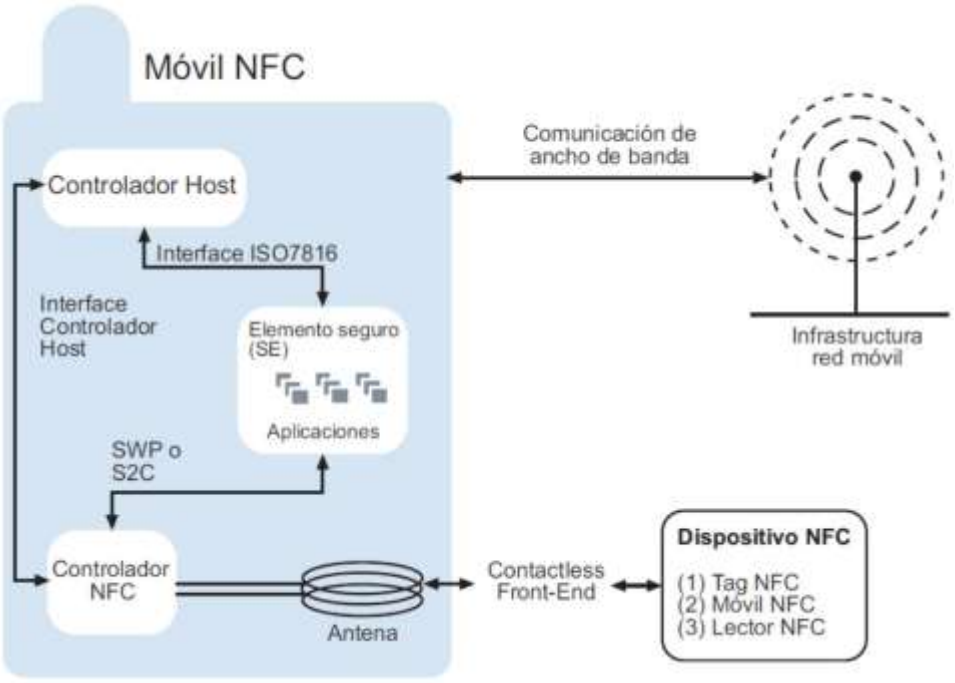


Figura 8 Elemento en un móvil NFC [25]

Los protocolos de intercambio de datos utilizados entre el SE y el controlador NFC son el SWP (Single Wire Protocol) y el NFC-WI (NFC Wire Interface). SWP se define en el estándar ETSI TS 102 613 y ETSI TS 102 622 y funciona con las siguientes características:

- Duplex,
- Hasta 1,6 Mbit/s,
- Transmisión de paquetes (orientado a bit).

Unos de los principales beneficios técnicos de SWP es lo que se conoce como "batery off". Si el teléfono se apaga o se acaba la batería, el SE puede ser alimentado por el interfaz del controlador NFC (usando la energía capturada por antena NFC). De esta manera, las aplicaciones sin contactos pueden estar disponibles incluso si la batería del terminal está vacía o si el teléfono está apagado. Esta característica es obligatoria para el transito masivo y el pago, ya que la gente tiene que poder usar los servicios NFC incluso si no tiene batería.

2.3.3 Papel de la SIM en el negocio móvil sin contacto

Los principales operadores de telefonía móvil recomiendan una arquitectura basada en SIM para los móviles sin contacto. La tarjeta SIM es el mejor lugar para alojar las aplicaciones NFC de una manera segura.

Los beneficios de la tarjeta SIM en el negocio de la NFC son:

- **Portabilidad y continuidad del servicio**. Cuando el cliente cambia de teléfono, mantiene todas sus credenciales (tickets, suscripciones, información de tarjeta de crédito, tarjetas de fidelización). Todas las aplicaciones asociadas están disponibles instantáneamente. No es necesario cargar ningún software o datos en el teléfono.

- **Particiones dedicadas para los bancos y los proveedores de servicios**. El proveedor de servicios tiene un control total sobre su partición en la tarjeta SIM y puede hacer su propio negocio independiente del operador de telefonía móvil. Este punto se verá más en detalle en la sección dedicada a Global Platform.

- **Control del operador móvil**. El operador de telefonía móvil emite la tarjeta SIM y es responsable de ello. Gracias a la tarjeta SIM, el operador está involucrado en el negocio de la NFC y por lo tanto, puede ofrecer servicios a terceros, tales como alquiler de espacios, gestión remota, etc

- **Administración remota.** La tarjeta SIM puede ser gestionada por el aire (OTA) a través de protocolos estándares existentes. Las aplicaciones de terceros podrían ser administrados por el Administrador de servicios de confianza (TSM) quien asumiría la seguridad y la autonomía también acorde con GlobalPlatform.

- **Seguridad**. La tarjeta SIM es un dispositivo de seguridad que puede ser utilizado para la banca, documentos de identidad como el pasaporte, el DNI etc. La certificación de las tarjetas SIM garantiza los altos niveles de integridad y seguridad requeridas por este tipo de aplicaciones.

2.4 GlobalPlatform

GlobalPlatform es una organización independiente, sin fines de lucro que se preocupa por tener una infraestructura estandarizada para el desarrollo, despliegue y gestión de tarjetas inteligentes. Fue fundada en 1999 para asumir la responsabilidad de la especificación Open Platform de Visa Inc que ya definió las bases de la autenticación, personalización y carga segura de aplicaciones sobre tarjetas bancarias a finales de la década de los 90. La aparición de tarjetas multi-aplicación (JavaCard) trajo consigo la necesidad de estandarizar los métodos de acceso y de encriptación. GlobalPlatform toma el testigo de esta especificación, convirtiéndose en un consorcio y produciendo la primera actualización en 2003.

A principios de 2010, GlobalPlatform ya contaba con 63 miembros.

La ejecución de las iniciativas GlobalPlatform se lleva a cabo por tres comités técnicos: el Comité de Tarjeta, el Comité de dispositivo y el Comité de Sistemas. A raíz de los planes de trabajo elaborados por el Consejo de Administración, estos tres grupos se encargan de formular los requerimientos del negocio y definir directrices para la implementación y el despliegue de los diferentes componentes de múltiples industrias.

La pertenencia a GlobalPlatform está abierta a cualquier organización interesada en la tecnología de tarjetas inteligentes incluidos los proveedores de hardware, chips y tecnología, asociaciones de pago, integradores, empresas de telecomunicaciones y agencias gubernamentales internacionales. Hay cinco niveles de pertenencia a GlobalPlatform - Integrante, Participante, Observador, Entidad Pública y Consultor - para satisfacer las preferencias de participación y los requisitos presupuestarios de las organizaciones. Una lista completa de los miembros actuales se pueden encontrar en el sitio web de GlobalPlatform http://www.globalplatform.org.

En los párrafos siguientes se hará un resumen de las diferentes especificaciones de GlobalPlatform que aplican en el ámbito del proyecto y se analizarán más en detalle los conceptos necesarios para comprender los mecanismos de seguridad empleados en la gestión de tarjetas multi-aplicación en un ecosistema NFC.

2.4.1 Especificaciones GlobalPlatform

La definición de las especificaciones GlobalPlatform se lleva a cabo por cada uno de los comités técnicos mencionados en la sección anterior, de tal manera que se clasifican por especificaciones de tarjeta, de dispositivo o de sistemas.

A nivel de tarjeta, existen varias versiones de una especificación principal, siendo el documento denominado GlobalPlatform Card Specification v2.2.1, la última versión publicada. Dicha especificación, define los componentes de la tarjeta, juegos de comandos, secuencias de transacción e interfaces que permiten la combinación de aplicaciones de cualquier industria en una sola tarjeta y que son neutrales a nivel de hardware, sistema operativo y proveedor. Todo ello en un marco de seguridad que se establece entre el emisor de tarjetas y los proveedores de aplicaciones. Para nuestro proyecto nos centraremos en los siguientes aspectos de la especificación:

- Arquitectura de la tarjeta
- Los protocolos, claves y mecanismos que garantizan una comunicación segura
- Los comandos y procesos para la gestión de contenido en la tarjeta, como es la carga, instalación y borrado de aplicaciones.

De entre las especificaciones de sistemas, encontramos la definición de los diferentes actores de un ecosistema NFC y los posibles modos para un despliegue comercial en el documento: GlobalPlatform Messaging for Mobile NFC Specification

2.4.2 Sistema Global Platform: Ecosistema NFC

Los principales actores involucrados en el despliegue de servicios móviles NFC y en su gestión son:

- **El Proveedor del Elemento seguro (SE Provider- Secure Element Provider):** el propietario del elemento seguro. En el caso de la SIM sería el operador de telefonía móvil.

- **El Proveedor de Servicios (SP – Service Provider):** una entidad como un banco, una autoridad de transporte, un minorista, etc. que proporciona servicios NFC a los consumidores. El SP necesita que sus servicios sean implementados en los equipos móviles de los usuarios finales.

- **El Operador de la Red Móvil (MNO – Mobile Network Operator)**: Proporciona la capacidad técnica para acceder al entorno móvil mediante un canal de comunicación OTA. El MNO puede ser también el proveedor de SE en caso de que el elemento de seguridad es la UICC..

- **El Gestor de servicios de confianza (TSM – Trusted Service Manager)**: un tercero de confianza, que proporciona uno o más roles técnicos y posiblemente roles de negocio a los otros actores. Los otros actores confían al TSM la gestión y la carga de los servicios. El TSM actúa como un agregador y puede soportar simultáneamente varios proveedores de servicio (SP) y varios operadores de redes móviles (MNO), manteniendo la confidencialidad entre los actores.

- **La Autoridad de Control (CA – Controlling Authority):** la entidad responsable de habilitar de manera segura la clave inicial en un dominio de seguridad GlobalPlatform.

Las responsabilidades respectivas de todos los actores no son las mismas para todos los despliegues móviles NFC. Las responsabilidades del proveedor de servicios SE y del TSM principalmente dependerá del acuerdo comercial que se establece para la gestión de contenidos de tarjeta como elemento seguro:

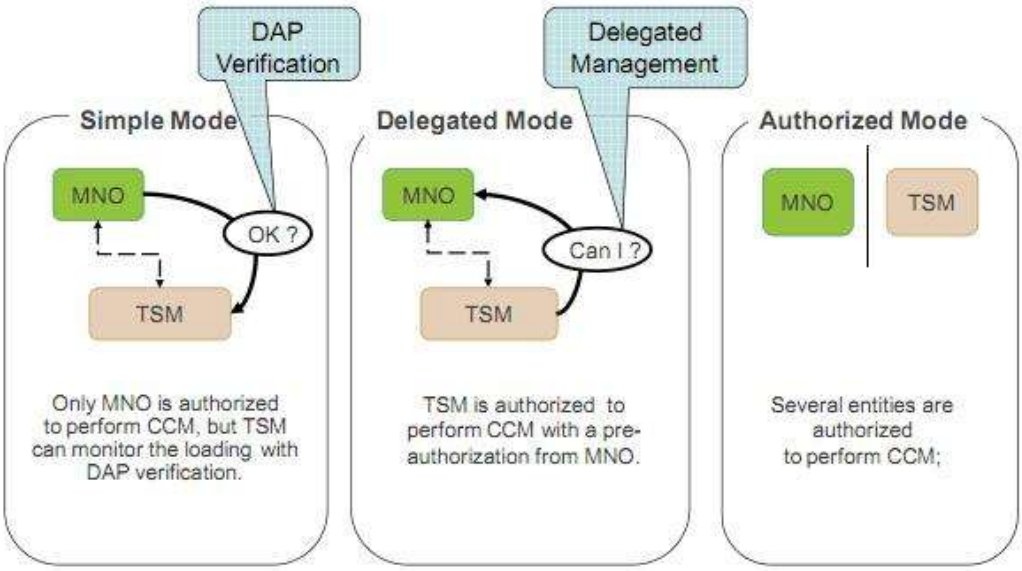


Figura 9 Modos de despliegue comercial NFC [2]

- **Modo simple**: un modelo centrado en el emisor, donde el TSM solicita al proveedor del elemento seguro SE, como por ejemplo el operador de telefonía móvil, realizar una operación de gestión de contenido de la tarjeta y devolver el resultado de la ejecución al TSM.

- **Modo delegado:** la gestión de contenido se delega en el TSM aunque cada operación requiere autorización previa del Proveedor del elemento seguro SE. El resultado de la ejecución de la operación se envía al proveedor del elemento seguro opcionalmente.

- **Modo dual o autorizado**: la gestión de contenido es totalmente delegada al TSM en un área específica del elemento seguro. El modo dual se caracteriza por la presencia de al menos dos dominios de seguridad con el privilegio de gestión autorizada en el elemento seguro.

Los acuerdos comerciales también pueden afectar la capacidad del TSM para comunicarse vía OTA con el elemento seguro. Dependiendo del modo implementado y / o las capacidades del elemento seguro:

- El TSM podría tener su propia capacidad de OTA para enviar los comandos de elemento seguro o de gestión de dispositivos por sí mismo.

- El TSM podría no tener ninguna capacidad de OTA. En este caso, el operador móvil o el Proveedor del elemento seguro SE proporciona un canal OTA para permitir al TSM enviar los datos o scripts que han sido previamente preparados por el mismo o el proveedor de servicios SP.

2.4.3 Tarjeta GlobalPlatform: Arquitectura (Dominio de Seguridad)

El principal propósito de la arquitectura de la tarjeta GlobalPlatform es garantizar un entorno de seguridad y gestión independiente entre los proveedores de aplicaciones y los sistemas de gestión de tarjetas. La siguiente figura muestra los componentes de una configuración de tarjeta que incluye una o más aplicaciones de la entidad emisora, una o más aplicaciones de los proveedores de servicios, y una o más aplicaciones de servicios globales (por ejemplo, servicios CVM) a otras aplicaciones.

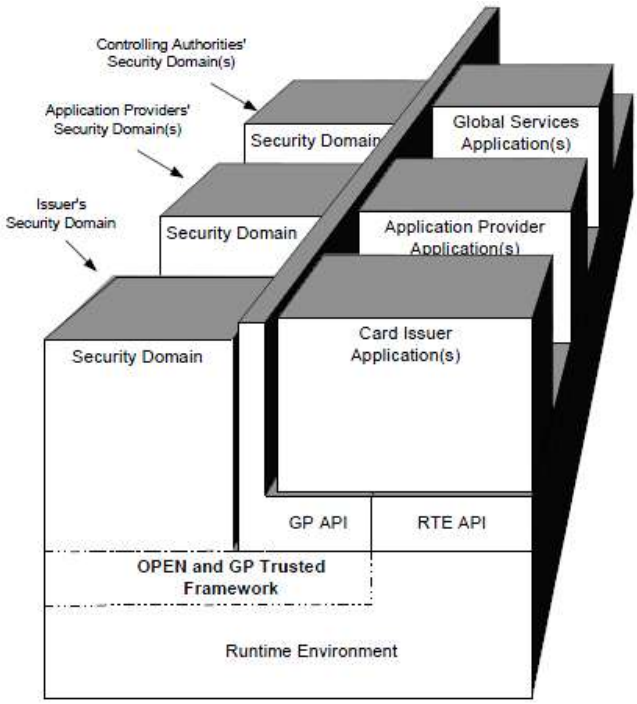


Figura 10 Arquitectura de tarjeta GlobalPlatform. [1]

En dicha arquitectura, cabe destacar la creación de aplicaciones de gestión de seguridad, llamadas Dominios de Seguridad, que aseguran una separación de claves completa entre el emisor de tarjetas y los múltiples proveedores de aplicaciones.

Los dominios de seguridad actúan como representantes en la tarjeta de las entidades fuera de ellas. Hay tres tipos principales de dominios de seguridad:

- **ISD** (Issuer Security Domain) o Dominio de Seguridad del Emisor: es obligatorio y se crea en representación del Administrador de la tarjeta, que típicamente es el emisor de la misma.

- **SSD** (Supplementary Security Domains) o Dominios de Seguridad Suplementarios: son opcionales y se crean en la tarjeta como representantes de los proveedores de aplicaciones, del emisor de la tarjeta, o de sus agentes (por ejemplo, agencias de servicio);

- **CASD** (Controlling Authority Security Domains) o Dominios de Seguridad de Autoridades de Control. La función de una Autoridad de Control es hacer cumplir la política de seguridad en todo el código de aplicación cargado en la tarjeta.

Los dominios de seguridad ofrecen servicios de seguridad tales como control de claves de cifrado, descifrado, generación y verificación de firma digital para sus proveedores de aplicaciones (Emisor de la Tarjeta, proveedor de aplicaciones o autoridades de control).

2.4.4 Tarjeta GlobalPlatform: Comunicación segura

La documentación GlobalPlatform define ampliamente la noción de comunicación segura por encima de los valores definidos en las normas ISO. Para ello, describe diferentes métodos de autenticación y de generación de sesiones mediante el establecimiento de canales seguros (Secure Channel), mecanismo indispensable para gestionar de manera segura los servicios proporcionados por los dominios de seguridad.

El empleo de canales lógicos facilita la posibilidad de tener varias entidades fuera de la tarjeta en comunicación simultánea con múltiples aplicaciones dentro de ella, cada una en su propio entorno lógicamente seguro. Es responsabilidad del proveedor de seguridad de dominio definir si esto es posible o no.

2.4.4.1 Canal Seguro

El canal seguro proporciona un canal de comunicación seguro entre una tarjeta y una entidad fuera de la tarjeta durante una sesión de aplicación. Una sesión de canal seguro se divide en tres fases secuenciales:

- **Inicio de Canal Seguro** – se establece cuando la aplicación en la tarjeta y la entidad fuera de ella han intercambiado información suficiente que les permita realizar las funciones criptográficas requeridas. La iniciación de una sesión de canal seguro siempre incluye la autenticación de la entidad fuera de la tarjeta por la aplicación en la tarjeta y viceversa.

- **Operación de canal seguro** - cuando la aplicación en la tarjeta y la entidad fuera de ella intercambian datos dentro de la protección criptográfica de una sesión de canal seguro. Los servicios ofrecidos por el canal seguro pueden variar de un **protocolo de canal seguro** a otro;

- **Terminación de canal seguro** - cuando la aplicación en la tarjeta o la entidad fuera de ella determina que ninguna otra comunicación se requiere o se permite a través de una sesión de canal seguro establecido.

2.4.4.2 Canal Seguro explícito / implícito

Existen dos maneras de iniciar una sesión de canal seguro:

- **Explícita**. Cuando una entidad fuera de la tarjeta inicia la sesión usando los comandos APDU apropiados o bien cuando es iniciada por una aplicación en la tarjeta usando un API determinado. Los comandos APDU permiten a la entidad fuera de la tarjeta determinar el nivel de seguridad requerido para llevar a cabo la sesión de canal seguro (integridad y/o confidencialidad). Estos comandos también proporcionan la posibilidad de seleccionar las claves que se van a utilizar.

- **Implícita**. El controlador del protocolo de canal seguro en la tarjeta inicia la sesión, directamente o a través de un API, cuando recibe el primer comando APDU que contiene una protección criptográfica. El controlador del protocolo de canal seguro conoce implícitamente el nivel de seguridad requerido y puede saber qué claves se van a utilizar. No soporta cifrado, únicamente integridad.

Tanto la aplicación en la tarjeta, mediante el API apropiado, como la entidad fuera de ella, mediante comandos APDU, pueden terminar la sesión de canal seguro. Cuando se termina la sesión, se reinician todos los datos asociados a dicha sesión.

2.4.4.3 Gestión directa / indirecta

Hay dos formas en las que la aplicación en la tarjeta puede manejar el protocolo de Canal Seguro:

- **Directa**: La aplicación posee su propio conjunto de claves e implementa plenamente el protocolo: un ejemplo es un dominio de seguridad.

- **Indirecta**: La aplicación utiliza los servicios del dominio de seguridad para manejar el protocolo de canal seguro. Esto permite que la aplicación que utiliza estos servicios se codifique de forma independiente del protocolo de canal seguro soportado por la tarjeta.

2.4.4.4 Entidad de autenticación

La entidad fuera de la tarjeta se autentica al iniciar una sesión de canal seguro. Si cualquiera de los pasos en el proceso de autenticación falla, el proceso se reinicia. La autenticación sólo es válida para esa sesión y sólo para los mensajes dentro de ese canal seguro. Si la sesión de canal seguro se cierra por cualquier razón la entidad fuera de la tarjeta dejará de ser considerada autenticada.

Existen dos tipos de protocolos criptográficos de autenticación:

- **Autenticación con Criptografía Simétrica**. Con un protocolo de clave simétrica (por ejemplo, SCP01 o SCP02), una entidad autenticada es aquella que conoce las claves secretas necesarias para iniciar una sesión de canal seguro. La aplicación es informada si el proveedor de la aplicación se ha autenticado correctamente o no examinando el indicador de autenticación del nivel de seguridad actual (véase la sección - Niveles de seguridad).

- **Autenticación con Criptografía Asimétrica**. Con un protocolo de clave asimétrica (por ejemplo, SCP10), cualquier entidad fuera de la tarjeta que posea un par de claves asimétricas y un certificado de la clave pública certificada por una autoridad reconocida por el dominio de seguridad puede ser autenticada correctamente por dicho dominio de seguridad.

2.4.4.5 Niveles de Seguridad

Un protocolo de canal seguro opera de acuerdo con el nivel de seguridad que se establezca. Esto se realiza a dos niveles:

- Un mínimo nivel de seguridad obligatorio que se establece al iniciar sesión ya sea explícita o implícitamente;
- Un nivel de seguridad diferente que se puede establecer para un comando o respuesta individual.

La siguiente tabla muestra la codificación de nivel de seguridad:

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
1	0	-	-	-	-	-	-	AUTHENTICATED
0	1	-	-	-	-	-	-	ANY_AUTHENTICATED
-	-	-	-	-	-	1	-	C_DECRYPTION
-	-	-	-	-	-	-	1	C_MAC
-	-	1	-	-	-	-	-	R_ENCRYPTION
-	-	-	1	-	-	-	-	R_MAC
-	-	-	-	X	X	-	-	RFU
0	0	0	0	0	0	0	0	NO_SECURITY_LEVEL

Tabla 2 Codificación de los niveles de seguridad. 1

Las reglas de cómo se manejan los niveles de seguridad son definidas individualmente para cada protocolo de canal seguro: SCP01, SCP02 o SCP10. En la siguiente sección se describe el protocolo SCP02, empleado en este proyecto.

2.4.5 Tarjeta GlobalPlatform: Gestión de contenido

La gestión de contenido en una tarjeta Global Platform (Card Content Management) es la capacidad de cargar, instalar, extraditar, actualizar el registro y borrar el contenido de la tarjeta.

Global Platform se diseña para proveer máxima flexibilidad al Emisor de Tarjeta (Card Issuer) y a sus socios de negocio. El diseño de Global Platform tiene en cuenta la posibilidad de que el Emisor de Tarjeta no tiene por qué querer necesariamente manejar todos los cambios de contenido, especialmente cuando el contenido de la tarjeta no

pertenece al Emisor. Por tanto, el Emisor de Tarjeta puede delegar la gestión de contenido a un Proveedor de Aplicaciones (Service provider) con o sin autorización:

- Puede autorizar todas las operaciones de gestión de contenido ejecutadas por el Proveedor de Aplicaciones
- Puede autorizar a un Proveedor de Aplicaciones tener el control total del contenido de su tarjeta
- Puede autorizar a un Proveedor de Aplicaciones aislar su propio Dominio de Seguridad y Aplicaciones de otros Proveedores de Servicio y potencialmente del Emisor de Tarjeta también.

Los cambios de gestión de contenido son permitidos de acuerdo a los privilegios que se asignan a los dominios de seguridad que existen en la tarjeta. Por ejemplo, un dominio de seguridad con el privilegio de gestión delegada, permite al proveedor de aplicaciones ejecutar:

- Carga de un aplicación delegada
- Instalación delegada
- Extraditación delegada
- Borrado delegado
- Actualización del registro de GlobalPlatform delegada

Es decir, necesitaría autorización del emisor de la tarjeta (el operador de telefonía móvil) o de un tercero para ejecutar dichas operaciones en la tarjeta. Normalmente esa autorización se traduce en el uso de tokens que se intercambian entre la partes y que tienen que añadirse a los comandos para su correcta ejecución, si no, el comando sería rechazado. En nuestro proyecto utilizaremos dominios de seguridad autorizados para simplificar el formateo de comandos que enviaremos a la tarjetas SIM.

La instalación de contenido en la tarjeta o bien se realiza simultáneamente con el proceso de carga, inmediatamente después del proceso de carga o en un momento posterior. En el primer caso el flujo de comandos es el mostrado en la Figura 11:

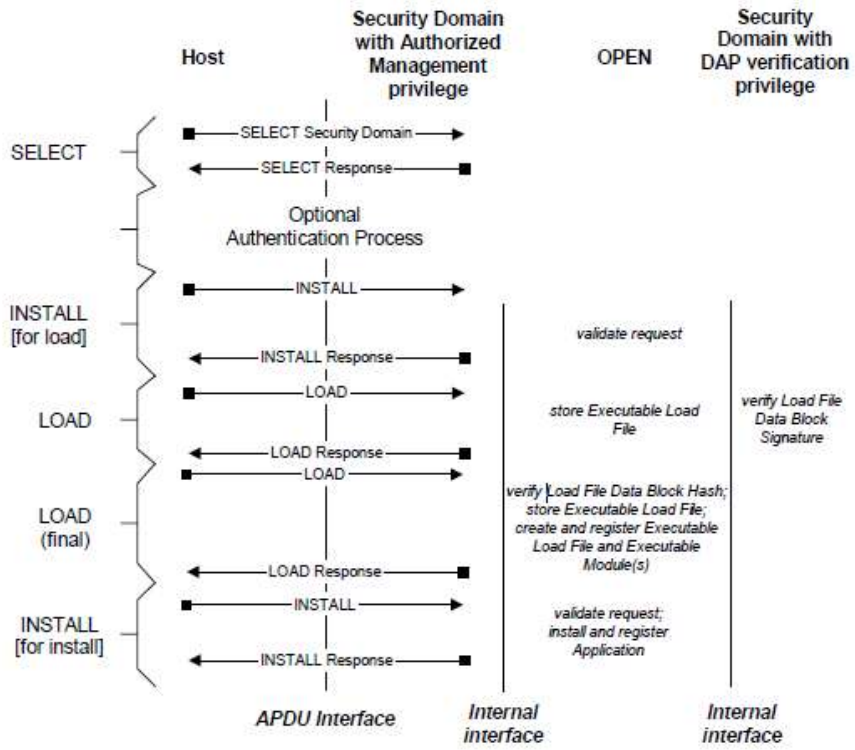


Figura 11 Diagrama de Carga e instalación de un servicio. [1]

Los comandos que utilizaremos durante el proceso de carga e instalación de contenido serán:

- **INSTALL**: El comando INSTALAR se expide a un dominio de seguridad para iniciar o realizar los diversos pasos necesarios para la gestión del contenido de la tarjeta.

Code	Value	Meaning
CLA	'80' - '8F', 'C0' - 'CF' or 'E0' - 'EF'	See section 11.1.4
INS	'E6'	INSTALL
P1	'xx'	Reference control parameter P1
P2	'00', '01' or '03'	Reference control parameter P2
Lc	'xx'	Length of data field
Data	'xxxx…'	Install data (and MAC if present)
Le	'00'	

Tabla 3 Estructura de comando INSTALL. [1]

El parámetro de control P1 define el papel específico del comando INSTALL, por ejemplo indicando si la aplicación va a ser cargada o instalada. .También permite que los datos de comando sean más de 255 bytes ya que son segmentados en componentes arbitrarios y transmitidos en una serie de comandos INSTALL. Se codifica de acuerdo a la siguiente tabla.

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	-	-	-	-	-	-	-	Last (or only) command
1	-	-	-	-	-	-	-	More INSTALL commands
-	1	0	0	0	0	0	0	For registry update
-	0	1	0	0	0	0	0	For personalization
-	0	0	1	0	0	0	0	For extradition
-	0	0	0	1	-	-	0	For make selectable
-	0	0	0	-	1	-	0	For install
-	0	0	0	-	-	1	0	For load

Tabla 4 Parámetro P1 de comando INSTALL. 1

- **LOAD**: Múltiples comandos de carga pueden ser utilizados para transferir un archivo a la tarjeta. El archivo de carga se divide en componentes más pequeños para la transmisión. Cada comando LOAD se enumera a partir de '00'. La numeración del comando será estrictamente secuencial y los incrementos por uno. La tarjeta será informada del último bloque del archivo de carga. Después de recibir el último bloque del archivo de carga, la tarjeta deberá ejecutar los procesos internos necesarios para la carga de archivos y los procesos adicionales identificados en el comando INSTALAR [de carga] que precedió a los comandos de carga.

Code	Value	Meaning
CLA	'80' - '8F', 'C0' - 'CF' or 'E0' - 'EF'	See section 11.1.4
INS	'E8'	LOAD
P1	'xx'	Reference control parameter P1
P2	'xx'	Block number
Lc	'xx'	Length of data field
Data	'xxxx..'	Load data (and MAC if present)
Le	'00'	

Tabla 5 Estructura de comando LOAD. 1

Capítulo 3

Tecnologías aplicadas

3.1 Introducción

En este capítulo se procede a explicar las tecnologías utilizadas en el desarrollo de la solución final, desde estándares web, elaboración de bases de datos con SQL, frameworks para el desarrollo de aplicaciones como Spring o tecnologías para la gestión remota de tarjetas MIFARE en entornos móviles como MIFARE4Mobile..

3.2 Estándares web

Con la aparición de Internet y de la web en concreto, se han abierto infinidad de posibilidades en cuanto al acceso a la información desde casi cualquier sitio. Este sistema de información es conocido como World Wide Web (WWW).

La web en sus orígenes fue pensada como un medio para desplegar información. Dicha información se encuentra contenida en servidores, denominados servidores web. La manera de acceder a las páginas web es a través de un navegador o browser, el cual realiza peticiones valiéndose del protocolo HTTP (HyperText Transfer Protocol).

La dirección que localiza la información dentro de Internet se denomina URL(Uniform Resource Locator) o Localizador Uniforme de Recursos.

Las características de la web son las siguientes:

- Global: Se puede acceder a ella desde cualquier tipo de plataforma, usando cualquier navegador y desde cualquier parte del mundo.
- Pública: Toda su información está distribuida en miles de ordenadores que ofrecen su espacio para almacenarla. Esta información es pública y toda puede ser obtenida por el usuario.
- Dinámica: La información, aunque esta almacenada, puede ser actualizada por quién la publicó sin que el usuario deba actualizar su soporte técnico.

La facilidad de comunicación que proporciona Internet conjuntada con la necesidad de acceso remoto a aplicaciones sin necesidad de instalaciones en la máquina del usuario ha hecho evolucionar este concepto. La comunicación ya no se basa simplemente en la carga de una página estática, sino que ésta puede ser el resultado de la ejecución en el servidor de alguna lógica de programación, es decir, interacción dinámica entre usuario y servidor.

La idea fundamental es que los navegadores, browsers, presentan documentos escritos en HTML, que han obtenido de un servidor web. Estos documentos HTML habitualmente presentan información de forma estática, sin más posibilidad de interacción con ellos.

El modo de crear los documentos HTML ha variado a lo largo de la corta vida de las tecnologías web pasando desde las primeras páginas escritas en HTML almacenadas en un fichero en el servidor web hasta aquellas que se generan al vuelo como respuesta a una acción del cliente y cuyo contenido varía según las circunstancias.

Así mismo, el modo de generar páginas dinámicas ha evolucionado, desde la utilización del CGI (Common Gateway Interface), donde se empleaban lenguajes de programación con sentencias de impresión para generar salidas HTML, hasta tecnologías tipo PHP, ASP o JSP, donde directamente se insertan pequeños fragmentos de lógica de programación en la estructura HTML de la página . Todas estas tecnologías se encuadran dentro de aquellas conocidas como Server Side, ya que se ejecutan en el servidor web.

Otro aspecto que completa el panorama son las inclusiones del lado del cliente, Client Side, que se refieren a las posibilidades de que las páginas lleven incrustado código que se ejecuta en el cliente, como por ejemplo *JavaScript* y programas Java (*Applets*).

3.3 HTML

HTML son las siglas de HyperText Markup Language (Lenguaje de Marcas de Hipertexto), es el utilizado para la construcción de páginas web. Su uso está destinado a describir la estructura y contenido de los sitios web en forma de texto, así como para complementar el texto con objetos tales como imágenes.

Este lenguaje se construye mediante etiquetas que delimitan las secciones de la web y alojan la información. Generalmente suele estar implementado junto con hojas de estilo llamadas CSS que ayudan a definir los aspectos visuales de la web.

Como se comentó anteriormente, HTML puede incluir scripts (JavaScript, PHP) que afectan al comportamiento de navegadores web y otros procesadores de HTML.

Puede ser creado y editado por cualquier editor de textos básico, como Gedit en Linux o Notepad++ en Windows, simplemente hay que tener en cuenta que la extensión del documento ha de ser *.htm o *.html. Dentro de estos archivos se implementará el código HTML el cual consta de varias etiquetas básicas:

- <**head**>: Define la cabecera del documento HTML, esta cabecera suele contener información sobre el documento que no se muestra directamente al usuario, como puede ser el título de la ventana del navegador.

- <**title**>: Define el título de la página que aparecerá encima de la ventana del sitio web.

- <**html**>: Define el inicio de un documento HTML, de esta forma se le indica al navegador que el texto que le sigue debe ser interpretado como código HTML.

- <**script**>: Introduce un script en una web, o se llama a uno mediante el atributo "*src= 'url del script'*".

- <**link**>: Para vincular el sitio a hojas de estilo o iconos.

- **<style>**: Coloca el sitio interno de la página, ya sea usando CSS u otros lenguajes similares.

- **<a>**: Para crear un enlace es necesario utilizar esta etiqueta de ancla junto al atributo href, que establecerá la dirección URL a la que apunta el enlace.

- **<body>**: Define el cuerpo o contenido principal del documento. Esta es la parte del documento que se muestra en el navegador. Dentro del cuerpo se pueden definir numerosas etiquetas, las más utilizadas son:

    - **<p>**: Define un párrafo de texto en la web.
    - **
**: Introduce un salto de línea
    - **<h1>**: Indica que se va a escribir una cabecera para una sección, el número indica el tamaño de la fuente
    - **<td>** : Define tablas en la página.
    - **<li>, <lu>** : Definen listas de elementos, ya sean ordenadas o no.

3.4 Hoja de estilos CSS

CSS o lo que es lo mismo, hojas de estilo en cascada, es un lenguaje utilizado para definir la presentación de un documento escrito en HTML o XML. El W3C es el encargado de formular las especificaciones de las hojas de estilo que servirán de estándar para los navegadores.

La idea que se mantiene para el desarrollo de las hojas de estilo es separar la estructura del documento de su presentación, lo que hace más fácil la modificación de los estilos en caso de ser necesario.

Cuando se utiliza CSS, las etiquetas no deben proporcionar información acerca de la visualización del documento, sino marcar la estructura. La información del estilo, separada en una hoja de estilo, especifica cómo se ha de mostrar una etiqueta; color, fuente, alineación del texto o tamaño.

La introducción de CSS ha permitido en muchos casos reemplazar el uso de tablas, que era la técnica que se utilizaba antes para conseguir el mismo fin. Sin embargo, CSS todavía no permite la misma versatilidad pues las diferencias entre los distintos navegadores dificultan la tarea. Se espera que futuros desarrollos en CSS3 resuelvan esta deficiencia y hagan de CSS un lenguaje más apto para describir la estructura espacial de una página. Como ventajas cabe destacar:

- Los navegadores permiten a los usuarios especificar la hoja de estilo local que será aplicada en un sitio web, con lo que se obtiene una mayor accesibilidad.
- Control centralizado de la presentación de un sitio web completo con lo que se agiliza la actuación del mismo de forma considerable.
- El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño.

3.5 PHP

El lenguaje PHP es un lenguaje de programación diseñado y orientado a la creación de páginas dinámicas. Es utilizado principalmente en interpretación del lado del servidor pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otro tipo de aplicaciones en las que se encuentran aquellas con interfaz gráfica.

La mayor fuerza de PHP reside en que es un lenguaje preparado para soportar accesos a muchos tipos de bases de datos como Oracle, ODBC, DB2, SQLServer...etc., y enfocando nuestro caso, MySQL. Lo que hace diferente a PHP es que el código que se deba ejecutar se ejecuta siempre en el servidor, de este modo el cliente solo recibe los resultados de la ejecución y le es imposible acceder al código que generó la página.

Figura 12 PHP

Las conexiones de PHP a bases de datos son enlaces SQL que no se cierran cuando termina la ejecución del script. El comportamiento de estas conexiones consiste en que al invocar una conexión, PHP comprueba si ya existe una conexión de este mismo tipo o es una conexión nueva. En el caso de que exista se procede a su uso y en caso que no exista la conexión se crea. Dos conexiones se consideran iguales cuando están realizadas al mismo servidor y con un mismo usuario y contraseña.

El código PHP se encuentra inmerso entre el código HTML y Java Script distinguido entre etiquetas:

<? php
código...
..... ?>

3.6 SQL

El lenguaje de consulta estructurado o SQL es un lenguaje declarativo de acceso a bases de datos relacionales que permite la especificación de diversos tipos de operaciones sobre éstas.

Actualmente es uno de los lenguajes de bases de datos más usado, tanto es así que SQL es el estándar que deben seguir los sistemas gestores de bases de datos relacionales.

Está considerado un lenguaje de alto nivel gracias a su fuerte base teórica y su orientación al manejo de conjuntos de registros, lo que permite una alta productividad en codificación y orientación a objetos. De este modo una única sentencia podría equivaler a uno o más programas que se utilicen en un lenguaje de bajo nivel.

Está compuesto por dos grandes sub-lenguajes con una temática bien diferenciada. Para la declaración de las estructuras y objetos de la base de datos se utilizará el Lenguaje de Definición de Datos (LDD) y para las fases de carga y consulta, SQL dispone del Lenguaje de Manipulación de Datos (LMD).

Para el lenguaje LDD existen cuatro operaciones básicas, todas ellas pueden actuar sobre una tabla, vista, índice, trigger, función, procedimiento o cualquier otro objeto que el motor de la base de datos soporte:

- CREATE: Este comando crea un objeto dentro de la base de datos.
- ALTER: Este comando permite modificar la estructura de un objeto.
- DROP: Este comando elimina un objeto de la base de datos. Se puede combinar con la sentencia ALTER.

3.7 Spring Framework

Spring es un framework para el desarrollo de aplicaciones y contenedor de inversión de control, de código abierto para la plataforma Java.

Este framework de desarrollo se ha utilizado para la creación del TSM en el proyecto Nubetrans debido a su alta flexibilidad obtenida por funcionalidades como la inversión de control (IoC en inglés) o la programación orientada a aspectos (AOP en inglés).

Spring es un framework que se originó en el año 2000 desarrollado por Rod Johnson, desde ahí ha ido evolucionando como un software de código abierto mantenido por un pequeño equipo de desarrolladores que a través del proyecto Sourceforge, lanzaron en el 2004 su versión 1.0.

Una de las metas de diseño Spring Framework es su facilidad de integración con los estándares J2EE y herramientas comerciales existentes, a su vez, Spring ha hecho llegar al gran público aquellas técnicas de desarrollo que hasta entonces resultaban desconocidas al gran público, técnicas como la inversión del control, donde el control de la ejecución no es lineal sino dinámico o el control orientado a aspectos.

Si bien las características fundamentales de Spring Framework pueden ser usadas en cualquier aplicación desarrollada en Java, existen variadas extensiones para la construcción de aplicaciones web sobre la plataforma Java EE. A pesar de que no impone ningún modelo de programación en particular, este framework se ha vuelto popular en la comunidad al ser considerado una alternativa, sustituto, e incluso un complemento al modelo EJB (Enterprise JavaBean).

Algunas de las características y/o módulos que el framework de Spring provee son los siguientes:

- Inversión de control (IoC)
- Programación Orientada a Aspectos (AOP)
- Acceso a datos
- Gestión de transacciones
- Modelo vista controlador
- Framework de acceso remoto
- Convención sobre configuración
- Procesamiento por lotes
- Autenticación y Autorización
- Administración Remota
- Mensajes
- Testing

3.8 REST

REST es el acrónimo de REpresentational State Transfer, o en español, Transferencia de Estado Representacional y define un estado de arquitectura de software para sistemas web.

REST se utiliza en la actualidad para describir la interfaz entre sistemas que utilizan directamente HTTP para obtener datos o indicar la ejecución de operaciones sobre los datos, en cualquiera de los formatos (XML, JSON, etc.) sin las abstracciones adicionales de los protocolos basados en patrones como puede ser SOAP. A estos sistemas que siguen los principios REST, se les denomina sistemas RESTful.

Las características principales de los sistemas REST son:

- Protocolo de comunicación cliente/servidor sin estado o "stateful" en el que camda mensaje que se intercambia entre cliente y servidor contiene toda la información necesaria para realizar la operación y ni el cliente ni servidor deben guardar ninguna información de la sesión, aunque en la práctica no siempre se sigue.
- Conjunto de operaciones bien definidas, utilizando las operaciones definidas en el estándar HTTP: PUT, GET, POST, DELETE.
- Sintaxis universal, cada recurso siempre se ejecuta sobre la misma URL

3.9 JSON

JSON es el acrónimo de JavaScrip Object Notation, es un formato de intercambio de mensajes ligero entre sistemas.

La simplicidad de JSON ha dado lugar a su extensivo uso en detrimento de XML sobre todo en AJAX o en el intercambio de mensajes entre dispositivos en los cuales el ancho de banda del canal es limitado como puede ser el mundo M2M o en dispositivos móviles.

El término JSON está altamente difundido en los medios de programación, sin embargo, es un término mal descrito ya que en realidad es solo una parte de la definición del estándar ECMA-262 en que está basado Javascript.

Otra de las ventajas de JSON contra otros Lenguajes de Marcado como XML es que al ser parta de Javascript y este estar presente en todos los navegadores web, su uso y análisis es extremadamente sencillo, situación que ha dado lugar a la alta extensión del formato entre la comunidad de desarrolladores.

Cada vez son más los lenguajes de programación que soportan mensajes JSON, entre ellos podemos encontrar los siguientes: ActionScript, C, C++, C#, ColdFusion,Common Lisp, Delphi, E, Eiffel, Java, JavaScript, ML, Objective-C, Objective CAML, Perl, PHP, Python, Rebol, Ruby, Lua y Visual FoxPro.

3.10 GCM (Google Cloud Messaging)

Como servicio de mensajería PUSH se ha decido utilizar Google Cloud Messaging o GCM.

Este servicio permite el envío de mensajes PUSH con un contenido de caracteres limitado a terminales Android o iOS, siendo en la plataforma Android un servicio integrado de base en el sistema operativo.

3.11 Protocolo SCP02

SCP02 es un protocolo de canal seguro de clave simétrica que permite mensajería de scripts a través de las interfaces de contacto y sin contacto de una tarjeta Global Platform. SCP02 provee los tres niveles siguientes de seguridad:

- Autenticación mutua - en la que la tarjeta autentica la entidad fuera de la tarjeta y viceversa, mediante el uso de claves simétricas;

- La autenticidad del origen y la integridad de los datos - en la que la entidad que recibe (la entidad fuera de la tarjeta o la tarjeta) asegura que los datos que se reciben en realidad procedían de la entidad autenticada, en la secuencia correcta y que no han sido alterados;

- Confidencialidad de los datos (cifrado) - en el que se asegura que los datos transmitidos desde la entidad emisora (la entidad fuera de la tarjeta o la tarjeta) a la entidad receptora (respectivamente la entidad fuera de la tarjeta o la tarjeta) no son visibles por una entidad no autorizada.

En SCP02 el parámetro "i" se define como un mapa de bits de la siguiente manera:

b8	b7	b6	b5	b4	b3	b2	b1	Description
Reserved							1	3 Secure Channel Keys
Reserved							0	1 Secure Channel base key
Reserved						1		C-MAC on unmodified APDU
Reserved						0		C-MAC on modified APDU
Reserved					1			Initiation mode explicit
Reserved					0			Initiation mode implicit
Reserved				1				ICV set to MAC over AID
Reserved				0				ICV set to zero
Reserved			1					ICV encryption for C-MAC session
Reserved			0					No ICV encryption
Reserved		1						R-MAC support
Reserved		0						No R-MAC support
Reserved	1							Well-known pseudo-random algorithm (card challenge)
Reserved	0							Unspecified card challenge generation method

Tabla 6 Codificación de parámetro i SCP02. 1

Por ejemplo, si el dominio de seguridad utiliza la implementación "i" = “55”, quiere decir que soporta modo explícito, C-MAC sobre APDU modificada, ICV configurado a cero, ICV cifrado para la sesión C-MAC, 3 Claves de Canal Seguro, Algoritmo pseudo-aleatorio conocido para el “Challenge” de tarjeta y no R-MAC.

Las sesiones seguras se basan en claves secretas compartidas entre la aplicación en la tarjeta y la entidad fuera de ella. Un dominio de seguridad deberá tener al menos un conjunto de 3 claves que se utilizan en la iniciación y el uso de un canal seguro. Son claves 3DES de 16 bytes de longitud y se nombran de la siguiente forma:

- **ENC**: clave para cifrar los datos del canal seguro.
- **MAC**: clave para calcular la MAC
- **DEK**: clave para cifrar datos sensibles, por ejemplo, claves secretas o privadas.

Estas claves se utilizan sólo para generar claves de sesión (S-ENC, S-MAC y S-DEK) durante la iniciación de un canal seguro.

3.11.1 Autenticación SCP02

La iniciación del Canal Seguro junto con la autenticación implica la creación de claves de sesión derivadas de claves estáticas usando un Contador de Secuencia mantenido por el Dominio de Seguridad. El Dominio de Seguridad gestiona un Contador por conjunto de claves.

El protocolo SCP02 describe tanto el proceso de autenticación explícita como el de implícita. En nuestro caso, el producto utilizado únicamente soporta autenticación explícita. El diagrama de la Figura 13 muestra el flujo de autenticación explícita de un canal seguro usando SCP02:

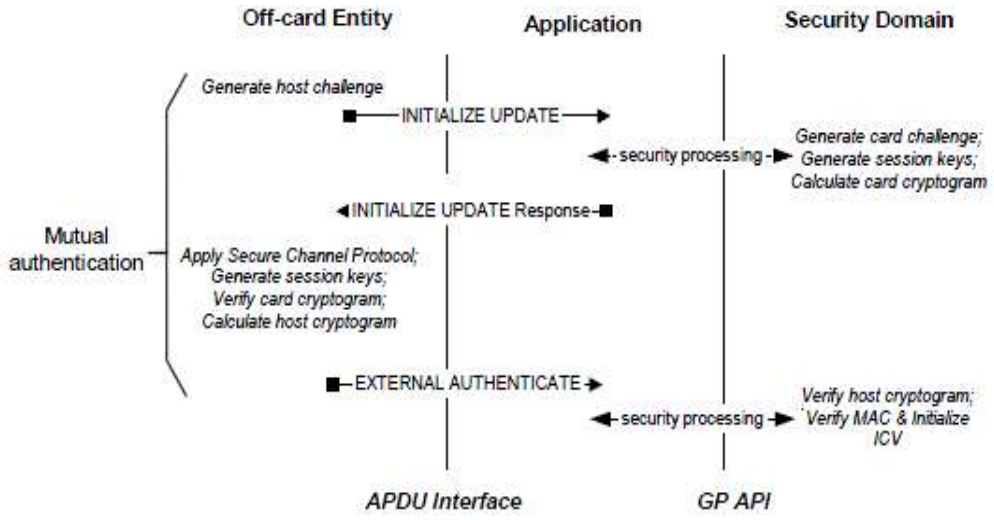


Figura 13 Flujo de autenticación explícita SCP02. 1

Los comandos empleados durante el proceso descrito anteriormente son:

- **INITIALIZE UPDATE**: es empleado para iniciar una canal seguro de manera explícita, donde se indica el conjunto de claves (dentro de un dominio de seguridad) que se usará para aplicar la seguridad al canal. Se codifica acorde a la tabla siguiente:

Code	Value	Meaning
CLA	'80' - '83	See section 11.1.4
INS	'50'	INITIALIZE UPDATE
P1	'xx'	Key Version Number
P2	'xx'	Key Identifier
Lc	'08'	Length of host challenge
Data	'xx xx…'	Host challenge
Le	'00'	

Tabla 7 Comando INITIALIZE UPDATE. 1

El mensaje de respuesta deberá contener la concatenación sin delimitadores de los siguientes elementos, donde "Key diversification data" será usado por el sistema final para derivar las claves estáticas de la tarjeta.

Name	Length
Key diversification data	10 bytes
Key information	2 bytes
Card challenge	8 bytes
Card cryptogram	8 bytes

Tabla 8 Respuesta de INITIALIZE UPDATE. 1

- **EXTERNAL AUTHENTICATE**: es utilizado por la tarjeta para autenticar el host y para determinar el nivel de seguridad requerido para todos los comandos subsiguientes. Una ejecución exitosa del comando UPDATE INITIALIZE precederá este comando.

Code	Value	Meaning
CLA	'84' - '87'	See section 11.1.4
INS	'82'	EXTERNAL AUTHENTICATE
P1	'xx'	Security level
P2	'00'	Reference control parameter P2
Lc	'10'	Length of host cryptogram and MAC
Data	'xx xx…'	Host cryptogram and MAC
Le		Not present

Tabla 9 Comando EXTERNAL AUTHENTICATE. 1

El nivel de seguridad se codifica en el P1 de la siguiente manera:

b8	b7	b6	b5	b4	b3	b2	b1	Description
0	0	0	0	0	0	1	1	C-DECRYPTION and C-MAC.
0	0	0	0	0	0	0	1	C-MAC
0	0	0	0	0	0	0	0	No secure messaging expected.

Tabla 10 Parámetro P1 del comando EXTERNAL AUTHENTICATE. 1

3.11.2 Claves de sesión

Las claves DES de sesión se generan cada vez que se inicia un canal seguro y puede ser usadas por los siguientes comandos si se requiere mensajería segura. Las claves de sesión se generan para garantizar que in conjunto de claves diferentes es usado en cada comunicación.

Las claves de sesión son el resultado de cifrado Triple DES en modo CBC de los datos derivados con la correspondiente clave estática inicial.

- La clave de sesión S-ENC es el cifrado Triple DES en modo CBC de los datos derivados con la clave ENC
- La clave de sesión S-MAC es el cifrado Triple DES en modo CBC de los datos derivados con la clave MAC
- La clave de sesión S-DEK es el cifrado Triple DES en modo CBC de los datos derivados con la clave DEK

Los 16 bytes de datos derivados se construyen de la siguiente manera:

Constante (2 bytes) + Contador de Secuencia (2 bytes) + Relleno ‘00’ (12 bytes)

Donde la constante es igual a:

- Para S-ENC: ’01 82’
- Para S/MAC: ’01 01’
- Para S-DEK: ’01 81’

Y el contador de secuencia se inicializa a ‘0000’ y se incrementa después de cada autenticación exitosa.

3.11.3 Desafíos y Criptogramas en Iniciación explícita

Tanto la tarjeta como la entidad fuera de la tarjeta (host) generan un desafío ("challenge") y un criptograma de autenticación. La entidad fuera de la tarjeta verifica el criptograma de la tarjeta y la tarjeta verifica el criptograma de acogida.

Los métodos de generación o verificación son los siguientes:

- **Desafío de Tarjeta (Card Challenge):** El desafío tarjeta es un número aleatorio o pseudo-aleatorio de 6 bytes que será único para una sesión de canal seguro.

- **Desafío de la entidad fuera de la tarjeta (Host Challenge):** El desafío de la entidad fuera de la tarjeta es un número aleatorio o pseudo-aleatorio de 8 bytes que será único para una sesión de canal seguro.

- **Criptograma de Tarjeta:** La generación y verificación del criptograma tarjeta se realiza utilizando la clave de sesión S-ENC y un ICV de ceros binarios, aplicando el método Triple DES al bloque de 24 bytes resultante de la concatenación de:

  8-byte Desafío anfitrión + 2-byte contador de secuencia + 6-byte desafío tarjeta + 8-bytes de relleno ('80 00 00 00 00 00 00 00 ').

  El resultado de esta operación es una firma de 8 bytes.

- **Criptograma de la entidad fuera de la tarjeta (Host):** La generación y verificación del criptograma anfitrión se realiza utilizando la clave de sesión S-ENC y un ICV de ceros binarios, aplicando el método Triple DES al bloque de 24 bytes resultante de la concatenación de:

  2-byte contador de secuencia, 6-byte desafío tarjeta, y 8-byte desafío anfitrión + 8-bytes de relleno ('80 00 00 00 00 00 00 00 ').

  El resultado de esta operación es una firma de 8 bytes.

3.11.4 MAC: Autenticidad e integridad del mensaje

El valor C-MAC es la firma de 8 bytes resultante de aplicar múltiples operaciones DES en cadena a través de la cabecera y el campo de datos del comando APDU con

relleno, sin incluir el campo Le. La entidad fuera de la tarjeta usará una clave de sesión MAC y un valor ICV durante su generación.

Para reflejar la presencia de C-MAC, la cabecera del comando APDU será modificada como sigue:

- La longitud del comando APDU (Lc) se incrementará en 8 para indicar la inclusión de la C-MAC en el campo de datos del mensaje de comando;
- El byte de clase será modificado para indicar que el comando APDU incluye mensajería segura. Esto se logra al configurar el bit 3 del byte CLA. Para todos los comandos definidos en GP 2.2.1, el valor del CLA será “84”.

Y para el cómputo de C-MAC,

- Los datos del comando APDU se rellenarán siguiendo la siguientes reglas:
    - Anexar un '80' a la derecha del bloque de datos;
    - Si la longitud del bloque de datos resultante es un múltiplo de 8 bytes, no se requiere más material de relleno;
    - Anexar ceros binarios a la derecha del bloque de datos hasta que la longitud de bloque de datos es un múltiplo de 8 bytes.

- El valor ICV dependerá de sobre qué APDU en la secuencia la MAC está siendo generada, de manera que se asegure la integridad de la secuencia de comandos:
    - Para el comando EXTERNAL AUTHENTICATE, el ICV se configura a ceros binarios;
    - Para cualquier otro comando que sigua al EXTERNAL AUTHENTICATE, el ICV es el valor C-MAC exitosamente verificado por el anterior comando recibido por la tarjeta.

El valor C-MAC se añade al final del comando APDU excluyendo cualquier relleno pero incluyendo las modificaciones realizadas en la cabecera del comando (clase y Lc).

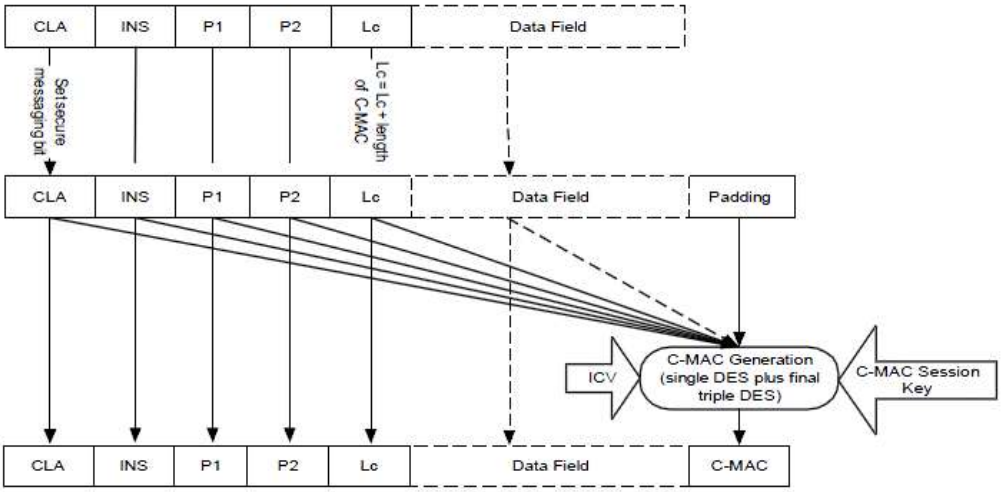


Figura 14 MAC SCP02. 1

3.11.5 Cifrado del mensaje

Si se requiere confidencialidad (cifrado) al iniciar un canal seguro, la entidad fuera de la tarjeta encripta el campo de datos del mensaje (comando APDU) que se transmite a la tarjeta. Esto excluye la cabecera y el C-MAC, pero incluye todos los datos dentro del campo de datos.

El cifrado y descifrado de comandos utiliza la clave de sesión S-ENC y Triple DES en modo CBC (definido en [ANSI X9.52] and [ISO 10116]) como método de cifrado, con un valor inicial de ICV de '00 00 00 00 00 00 00 00'.

Antes de la encriptación, los datos se rellenan de acuerdo a la reglas de relleno DES siguientes y esto requiere la modificación adicional del valor Lc.

- Anexar un '80' a la derecha del bloque de datos;
- Si la longitud del bloque de datos resultante es un múltiplo de 8 bytes, no se requiere más material de relleno;
- Anexar ceros binarios a la derecha del bloque de datos hasta que la longitud de bloque de datos es un múltiplo de 8 bytes.

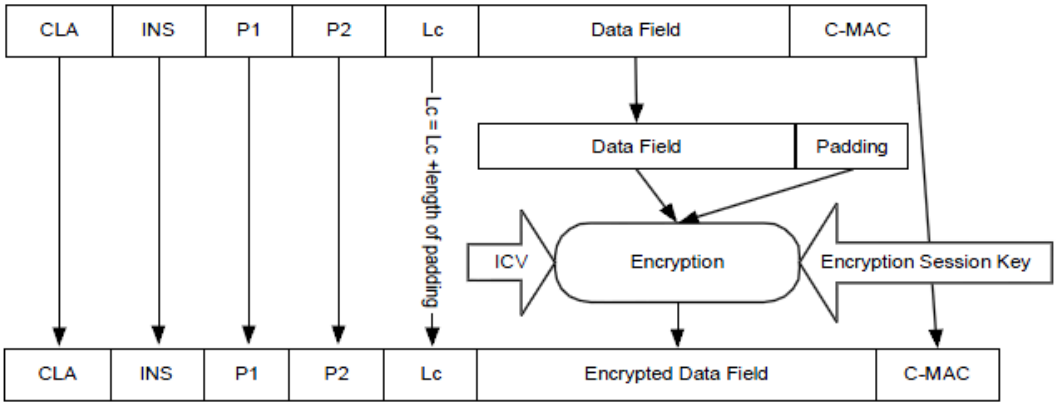


Figura 15 Cifrado SCP02 de Comando de datos APDU. [1]

3.12 MIFARE4Mobile

MIFARE4Mobile V1.01 [3] es una tecnología que provee una solución punto a punto para gestionar servicios MIFARE Classic en un Elemento Seguro (UICC) integrado en un móvil NFC. Los servicios MIFARE Classic son almacenados en el área de Memoria MIFARE de la UICC, por tanto, pueden tener la misma funcionalidad sin contacto que las tarjetas MIFARE Classic convencionales. Cada servicio MIFARE puede ocupar uno o más sectores. Sin embargo, cada sector solo puede albergar un servicio particular MIFARE.

La distribución remota de aplicaciones MIFARE a dispositivos habilitados para ello vía redes GSM/3G es un requisito importante para varios mercados existentes que han desplegado infraestructuras MIFARE y que han emitido tarjetas MIFARE a los usuarios de dicha infraestructura. El propósito en un entorno NFC es que los usuarios/consumidores utilicen las capacidades adicionales de MIFARE en los teléfonos móviles para aumentar la utilidad del servicio que ofrecen dichas tarjetas. Entre las capacidades extendidas se encuentran:

- Visualizar el contenido de la tarjeta y el historial de transacciones desde la interfaz de usuario del teléfono.
- Recargar la aplicación con adicionales derechos, créditos o características.
- Habilitar y deshabilitar los servicios cargados o
- descargar nuevos servicios

MIFARE4MOBILE es una tecnología desarrollada por NXP para satisfacer estos requerimientos. A fin de asegurar la interoperabilidad entre los proveedores de servicio MIFARE y las implementaciones de MIFARE4Mobile en SIMs y SEs (Secure Elements) de teléfonos móviles NFC, ciertos interfaces claves deberían ser definidos y acordados en todo el sector de tal manera que la adopción de servicios basados en NFC MIFARE no se vea comprometida

En los siguientes párrafos se describirá en la arquitectura MIFARE4Mobile v1.01 en un entorno NFC con tos sus componentes y funcionalidades.

3.12.1 Arquitectura en un entorno MIFARE4Mobile

El siguiente diagrama muestra el entorno MIFARE4Mobile de un teléfono móvil NFC:

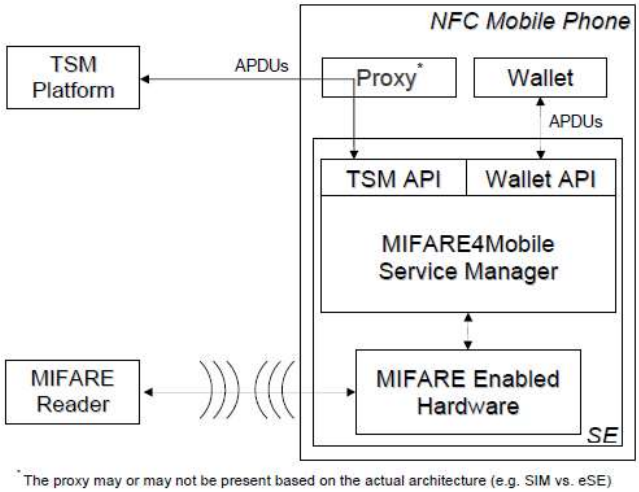


Figura 16 Entorno MIFARE4Mobile en un Teléfono móvil NFC. 3

Veamos las diferentes partes de este entorno en detalle.

- **Elemento Seguro (SE – Secure Element):**

  El elemento seguro podría ser una UICC (Universal Integrated Circuit Card), un SE embebido u otro tipo de SE desmontable como uSD cards.

- **MIFARE4Mobile Gestor de Servicio (SM - Service Manager):**

  El Service Manager de MIFARE4Mobile gestiona el ciclo de vida de la aplicación MIFARE en los móviles NFC. La aplicación MIFARE es una

colección de datos que pueden ser cargados en la memoria de un dispositivo habilitado para MIFARE. Para ello dispone de dos APIs: el TSM API y el Wallet API. El API para el TSM permite emitir, anular o actualizar aplicaciones MIFARE mientras que el API para el Wallet permite recuperar información sobre las aplicaciones MIFARE ya emitidas. Además, el Service Manager de MIFARE4Mobile tiene acceso al hardware habilitado para MIFARE en el móvil NFC con el fin de leer y escribir aplicaciones MIFARE.

El Service Manager de MIFARE4Mobile debe ser compatible con la especificación de Global Platform 2.1.2 y estará ubicado dentro del elemento seguro de los teléfonos móvil NFC.

- **PROXY:**

  El Proxy se utiliza para establecer una conexión segura entre la plataforma del TSM y el Service Manager de MIFARE4Mobile cuando el elemento seguro (SE) no tiene capacidades de gestión remota (OTA) o se decide no utilizarlas en favor de este método independiente de la subscripción móvil.

- **PLATAFORMA TSM:**

  La plataforma del TSM recibe las peticiones de los Proveedores de Servicio (emisiones, anulaciones y actualizaciones). Los principales componentes del TSM es un módulo que almacena de forma segura las claves NFC y un generador de comandos. El TSM es responsable de:

  1. Implementar y ejecutar las políticas del Emisor de Tarjetas
  2. Instalar el Service Manager de MIFARE4Mobile y mantener sus claves, generadas por el Proveedor de Servicios, siguiendo las políticas del Emisor de Tarjetas.
  3. Recuperar las claves MIFARE4Mobile del fabricante del elemento seguro y cargarlas
  4. Hacer cumplir los estándares y políticas de los Proveedores de Servicio que rigen todos los aspectos de las aplicaciones MIFARE4Mobile
  5. Trabajar con los Proveedores de Servicio para crear e inicializar las aplicaciones MIFARE4Mobile
  6. Determinar las políticas en relación con la tarjeta y la gestión del ciclo de vida de las aplicaciones MIFARE4Mobile.
  7. Manejar la carga del código de la aplicación y la instalación tanto en fábrica (pre-emisión) como en tarjetas ya emitidas.

- **TSM API:**

El TSM API provee funciones a la plataforma del TSM para manejar el ciclo de vida de las aplicaciones MIFARE (emisión, actualización, recarga o anulación) y las claves de seguridad. Existen dos puntos a nivel de seguridad que deben ser gestionados:

- Una autenticación mutua de acuerdo a Global Platform, que debe establecerse con MIFARE4Mobile Service Manager para acceder al TSM AP
- Todos los datos transmitidos entre la plataforma del TSM y el Service Manager de MIFARE4Mobile debería estar cifrada.

A continuación se expone una breve descripción de las funciones disponibles en el TSM API, que serán utilizadas en la implementación del proyecto NUBETRANS.

API	APDU	Descripción
Initialize Update	INITIALIZE UPDATE	Inicia la autenticación obligatoria
External Authenticate	EXTERNAL AUTHENTICATE	La autenticación es el paso obligatorio para acceder al resto de las funciones del TSM API
Delete	DELETE	Anula una aplicación MIFARE4Mobile ya emitida. La aplicación debe ser borrada tanto de la memoria Java Card como de la memoria MIFARE.
Install [for load]	INSTALL	Verifica si una aplicación MIFARE4Mobile puede ser emitida. Precede al comando LOAD.
Install [for reload]	INSTALL	Verifica si es posible una actualización de los metadatos de una aplicación MIFARE4Mobile ya emitida. Precede al comando LOAD.
Install [for install]	INSTALL	Finaliza la emisión/ recarga de una aplicación MIFARE4Mobile. Sigue a los comandos LOAD
Load	LOAD	Carga datos y/o metadatos de una aplicación MIFARE4Mobile en el applet. Dependiendo del tamaño de la aplicación, varios LOAD pudieran ser necesarios.
Put Key	PUT KEY	Actualiza las claves de autenticación del MIFARE4Mobile Service Manager.
Put MIFARE Keys	PUT MIFARE KEYS	Inicializa o actualiza las claves de la memoria MIFARE
Update Data	UPDATE DATA	Actualiza los datos de la aplicación MIFARE de

		una aplicación MIFARE4Mobile ya emitida
Get Data	GET DATA	Recupera información específica del TSM y especialmente el UID de MIFARE.
Store Data	STORE DATA	Almacena información específica del TSM
Unblock PIN	UNBLOCK PIN	Formatea el contador de tanteos del PIN, desbloquea el MIFARE4Mobile Service Manager y cambia opcionalmente el PIN.

Tabla 11 Funciones del TSM API MIFARE4Mobile. . 3

- **WALLET:**

  El Wallet es el interfaz gráfico entre el usuario final y el Service Manager de MIFARE4Mobile. La tecnología usada para el Wallet depende de la arquitectura del teléfono móvil NFC.

- **WALLET API:**

  Como se comentó anteriormente, el Wallet API provee una serie de funciones que permiten recuperar información sobre las aplicaciones MIFARE que ya han sido emitidas y mostrarla por el interfaz de usuario del teléfono. La descripción detallada de las funciones del Wallet API está fuera del alcance de este proyecto, pero se puede obtener más información en la especificación "MIFARE4Mobile Interface specification v1.01" 3 .

- **Lector MIFARE (MIFARE Reader):**

  El lector MIFARE accede al Hardware habilitado para MIFARE en el teléfono móvil NFC.

3.12.2 Implementación MIFARE4Mobile 1.01

La implementación de MIFARE4Mobile v1.01 gira entorno a dos conceptos claves: el Gestor de Servicio y el Objeto de Servicio.

3.12.2.1 MIFARE4Mobile: Definición de Gestor de Servicio

El Gestor de Servicio (Service Manager) de MIFARE4Mobile es responsable de:

- Recibir y tratar los comandos APDUs que llegan del TSM o del Wallet:
  - El Gestor de Servicio maneja la autenticación,
  - Verifica el nivel de seguridad, la MAC si está presente y descifra los datos cuando es necesario
  - Ejecuta los comandos recibidos
  - Cifra la respuesta cuando es necesario y envía de vuelta.

- Manejar el contenido de los datos
  - El Gestor de Servicio permite al TSM la personalización y actualización de la memoria MIFARE mediante el almacenaje de datos propietarios
  - Puede cargar, instalar, actualizar o anular los objetos del servicio MIFARE4Mobile, y por lo tanto maneja el ciclo de vida completo de los mismos.

3.12.2.2 MIFARE4Mobile: Definición de Objeto de Servicio

Antes de nada, aclarar que cada tarjeta de transporte de Nubetrans cargada en una SIM NFC se corresponde con un Objeto de Servicio MIFARE4Mobile. La versión V1.01 de MIFARE4Mobile sólo soporta aplicaciones MIFARE Classic 1K. En dicho contexto, un Objeto MIFARE4Mobile está compuesto de los siguientes elementos:

- Datos: incluyen las claves y las condiciones de acceso a la memoria MIFARE.
- Metadatos privados: incluyen algunas variables que se definirán más adelante como “Sector offset”, “MAD ID”, “Parser ID”, “Config URL” y contenido propietario.
- Metadatos públicos: incluyen las variables “Label”, “SOID” y “Dates” que también se definirán más adelante.

Figura 17 MIFARE4Mobile Objeto de Servicio. . 3

Para definir un Objeto de Servicio se utiliza codificación TLV

1. **TAG:** campo de 1 byte de longitud que identifica una variable del objeto MIFARE4Mobile. Ver la tabla
2. **Longitud**: campo de 1, 2 o 3 bytes de longitud que contiene el tamaño del campo "Valor". Existen algunas limitaciones descritas en la especificación "MIFARE4Mobile Interface Specification v1.01".
3. **Valor**: campo de 1 a 65535 bytes de longitud que contiene información acorde con la Tabla 12:

Variable	Presencia	TAG	Longitud	Valor
Label	Obligatoria	'81'	Hasta 32	Etiqueta asociada a la aplicación MIFARE. Debe ser codificada en UTF8
SOID	Obligatoria	'82'	9	Define un único valor para cada aplicación MIFARE cargada en el Service Manager
Begin Date	Opcional	'83'	9	Fecha de inicio de validez de la aplicación.
End Date	Opcional	'84'	9	Fecha de fin de validez de la aplicación.
Sector offset	Obligatoria	'85'	2	Sector de la memoria MIFARE donde la aplicación es cargada.
Data	Obligatoria	'86'	Hasta 65535	Array de bytes que representa los sectores de datos de la aplicación MIFARE excluyendo el sector "trailer". La representación utiliza pares de longitud-datos de la forma: 48<…48 bytes de datos del primer sector>48<…48 bytes de datos del segundo sector >
Keys	Obligatoria	'87'	Hasta 65535	Array de bytes que representa los sectores trailers KEY A y KEY B de la aplicación MIFARE, excluyendo el sector tráiler con los bits de acceso.
Access Conditions	Obligatoria	'88'	Hasta 255	Array de bytes con las condiciones de acceso de la aplicación MIFARE.
MAD ID	Opcional	'89'	2	
Parser ID	Opcional	'8A'	16	
Config URL	Opcional	'8B'	Hasta 127	
Propietary Content	Opcional	'8C'–'9E'	Hasta 65535	

Tabla 12 Codificación TLV MIFARE4Mobile Objeto de Servicio. . 3

El ciclo de vida de un Objeto de Servicio tiene los siguientes estados:

- **CARGADO**: significa que los datos y los metadatos del objeto se han cargado en el Gestor de Servicio.
- **ACTIVADO**: significa que el objeto se ha creado y que es visible para un lector NFC, es decir, que está presente en la memoria MIFARE de la SIM.
- **DESACTIVADO**: significa que el objeto se ha creado pero que no es visible para un lector NFC.

3.12.2.3 MIFARE4Mobile: Comunicación Segura

Es exactamente el mismo proceso definido en la especificación de Global Platform de Tarjeta, tanto para la autenticación como para la carga de los datos. En el caso de Nubetrans, como se mencionó anteriormente, se utilizara el protocolo SCP02 con nivel de seguridad ‘03’, que significa que los datos se enviarán cifrados y con MAC una vez iniciado el canal.

Capítulo 4

Descripción general de la solución

4.1 Introducción

Nubetrans es un proveedor de servicios que ofrece una solución remota de gestión de títulos de transporte basada en teléfonos con capacidad NFC. La idea consiste en una aplicación web mediante la cual cualquier persona con un móvil y una SIM NFC que soporte Mifare4Mobile, pueda comprar y recargar billetes que residan en su teléfono móvil y que den acceso a la red de diferentes operadores de transporte.

El cliente no tendrá que desplazarse a ningún punto de venta físico ni tendrá que esperar colas para recargar su billete antes de coger el tren, metro o el autobús de cualquiera de los socios de transporte de Nubetrans. Tampoco tendrá que preocuparse de llevar consigo ningún billete impreso, abono transporte o dinero para moverse por su ciudad sino que gestionará sus billetes en el móvil desde cualquier ordenador o dispositivo con conexión a internet en el momento que le sea más conveniente.

Para utilizar los servicios ofrecidos por Nubetrans, el usuario tendrá que registrarse en la web y dar su consentimiento para la gestión remota de los billetes en su tarjeta SIM NFC. Durante el registro, Nubetrans creará un perfil nuevo de usuario en su base de datos junto con los credenciales, login y contraseña, que le darán acceso a dicho perfil. También pedirá al cliente que se descargue una aplicación Android en el móvil para consultar los títulos disponibles y para gestionar los mensajes enviados desde la aplicación web.

Una vez registrado, el cliente podrá comprar, recargar, consultar o cancelar electrónicamente cualquier título de transporte disponible en su perfil de manera privada y segura. Para realizar dichas operaciones, la aplicación web llamará al servidor de aplicaciones, el TSM en funciones, encargado de formatear los comandos y añadir la seguridad al mensaje que se enviará al móvil NFC. La aplicación Android recibirá el mensaje y se lo mandará a la tarjeta SIM, encargada de gestionar la seguridad y de ejecutar los comandos para hacer efectiva la operación solicitada por el usuario.

El presente capítulo presenta una visión global del sistema Nubetrans.

4.2 Funcionalidad

La web de Nubetrans está disponible en cualquier navegador a través de la URL http://www.nubetrans.com. Al entrar veremos la página de inicio (Figura 3.1) donde el usuario podrá registrarse o iniciar una sesión con su perfil si ya está registrado para llevar a cabo la compra, consulta, recarga o cancelación de sus títulos de transporte.

Figura 18 Página de Inicio Nubetrans

Todas las páginas HTML que se muestran tienen una cabecera común con cuatro pestañas:

- **Inicio**: Carga la página principal (Figura 3.1)
- **Servicios**: Muestra los operadores de transporte en España que están disponibles en la web de Nubetrans para la gestión de sus billetes NFC en el móvil.
- **Billete NFC:** Explica al cliente los requisitos para poder utilizar el móvil como medio de pago y acceso a la red de transportes. También explica el proceso de emisión y carga del billete NFC en el móvil.
- **Atención al cliente**: Muestra los números de atención telefónica y horarios para dar soporte al cliente en caso de dudas, reclamaciones o para aceptar sugerencias.

Una vez que el usuario haya iniciado sesión y este conforme con la compra o recarga solicitada, se creará un petición HTTP POST sobre REST utilizando un mensaje JSON con los datos de servicio Mifare seleccionado. Dicha petición se enviará a la plataforma del TSM, en este caso una aplicación alojada en un servidor de aplicaciones, que construirá el script con la seguridad pertinente y lo enviará al móvil NFC.

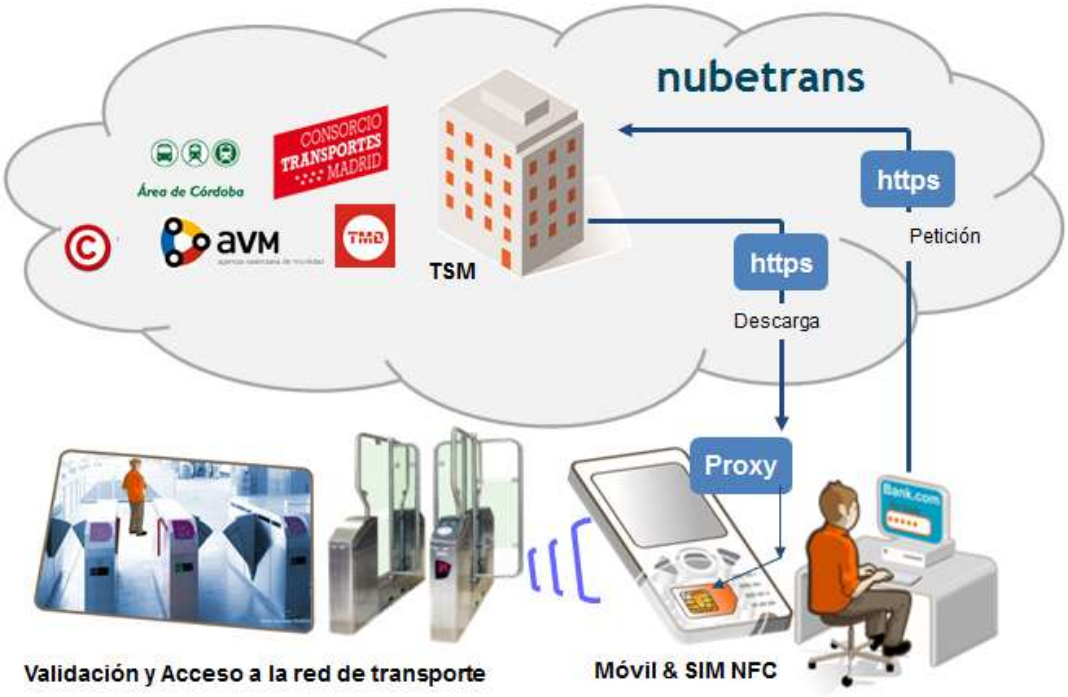


Figura 19 Peticiones HTTP(S) desde Nubetrans

4.3 Requisitos

El objetivo de esta sección es recoger los requisitos que se han de satisfacer para poner en marcha el sistema. Se distinguirá en dos roles concretos: el de desarrollador y el de usuario final de la aplicación.

4.3.1 Requisitos del desarrollador

Para desarrollar e implantar este proyecto se hacen indispensables los siguientes elementos:

- **XAMPP v5.5.24/5.6.8** [23]

  Para montar la página web en local y tener tener Apache, MySQL, PHP y phpMyAdmin en la máquina donde hemos desarrollado la aplicación nos hemos descargado la versión 5.5.24/5.6.8 de XAMP para Windows en https://www.apachefriends.org/es/index.html.

- **MySQL Workbench 5.2**

  Para desarrollar y administrar la base de datos, nos hemos instalado MySQL Workbench, una herramienta que ofrece 3 interfaces gráficas para facilitar las consultas a la base de datos, el diseño y la definición la arquitectura de la misma y la administración de la base de datos y del servidor en general.

- **Eclipse 4.5**

  Para el desarrollo del TSM, nuestra aplicación Java 1.8 que recibe las peticiones de la aplicación web y que se las envía al móvil tras formatearlas de acuerdo al estándar Mifare4Mobile v1 y al esquema de seguridad SCP02, se ha utilizado el IDE Eclipse 4.5, en su versión Mars. Para el desarrollo de la aplicación Android también se ha utilizado el paquete ADT (Android Development Tools) y el Plug-in de Spring para Eclipse. Todo ello disponible en la web de Eclipse (http://www.eclipse.org/downloads/).

- **Gemalto Mifare Demo Tool 3.0.0**

  Para validar la correcta personalización de la memoria MIFARE en la tarjeta SIM tras una petición de gestión del título de transporte (compra, recarga etc) lanzada por la web de Nubetrans, La herramienta Mifare Demo permite probar la

comunicación ISO 14443-3 RF entre el lector y el CLF y muestra el resultado de leer uno o más sectores Mifare.

- **Lector de tarjetas Dual Gemalto Prox –DU.**

  Es un lector de tarjeta inteligente de doble interfaz que se conecta a entornos de PC y otros sistemas utilizando el interfaz USB 2.0 de máxima velocidad. Conveniente tanto para aplicaciones que utilizan la tecnología sin o con contacto de la tarjeta inteligente. En nuestro caso se utilizará tanto para la creación de la tarjeta SIM mediante el interfaz con contactos como para la validación de la carga del título de transporte en la tarjeta SIM. Para este segundo paso, se acercará el móvil a la interfaz sin contactos del lector previamente seleccionada en una aplicación PC Flash que simulará la entrada al metro (véase Figura 3.2)

Figura 20 Lector Dual Gemalto Prox –DU. 15

- **Terminal móvil con soporte NFC.**

  En este proyecto hemos utilizado un LG Optimus L7 II (LG P710) como el que se puede ver en la Figura 3.3.

Figura 21 LG L7 II

- **Gemalto WinSPI**

  WinSPI es una herramienta PC de Gemalto que permite ejecutar scripts paso a paso sobre una tarjeta SIM. En nuestro caso la utilizaremos para depurar el script Mifare4Mobile con seguridad que formatea nuestra aplicación web con el objeto de crear un billete NFC en la tarjeta SIM. La comunicación entra la herramienta y la tarjeta se hará a través del interfaz con contactos del lector Gemalto Prox-DU.

- **Tarjetas Inteligentes**

  En este proyecto hemos utilizado tarjetas inteligentes Gemalto STM041 con la versión mNFC 2.1 (Java Card 2.2.2 & GP 2.2 API). También será necesario un perfil de tarjeta SIM concreto que incluya las siguientes aplicaciones Java Card:

  - Mf4M1.01_core-GP22-Interop_ReleaseA.2_250712. **16** Es el principal paquete Java que implementa Mifare4Mobile v1.01. Es el Gestor de Servicio MIFARE (Service Manager) dentro de la tarjeta y provee los API para la comunicación con el wallet (en nuestro caso la aplicación Android) y con el TSM.

  - Mf4M1.01_CodecInterface_core-GP22 Interop_ReleaseA.2_250712. **16.** Es un interfaz que permite al códec decodificar el mapeo de datos MIFARE.

  - Códec/Parser del Proveedor de Servicios. El códec es la aplicación que se utiliza para codificar y decodificar la información específica del servicio MIFARE, como el saldo de la cuenta.

  - Mifare Companion. El Mifare companion es la entidad que tiene acceso directo a la emulación MIFARE en la tarjeta SIM y gestiona el intercambio de servicios Mifare4Mobile. Contiene todos los objetos de servicio y mantiene la relación entre ellos y sus respectivos dueños (proveedores de servicios) en este caso Nubetrans.

  La figura 3.4 muestra la distribución de las aplicaciones mencionadas en una estructura de dominios de seguridad autorizada, la cual permite delegar la gestión de contenido al TSM sin necesidad de consentimiento del operador de telefonía, simplificando así el mensaje que se tendrá que formatear en el TSM.

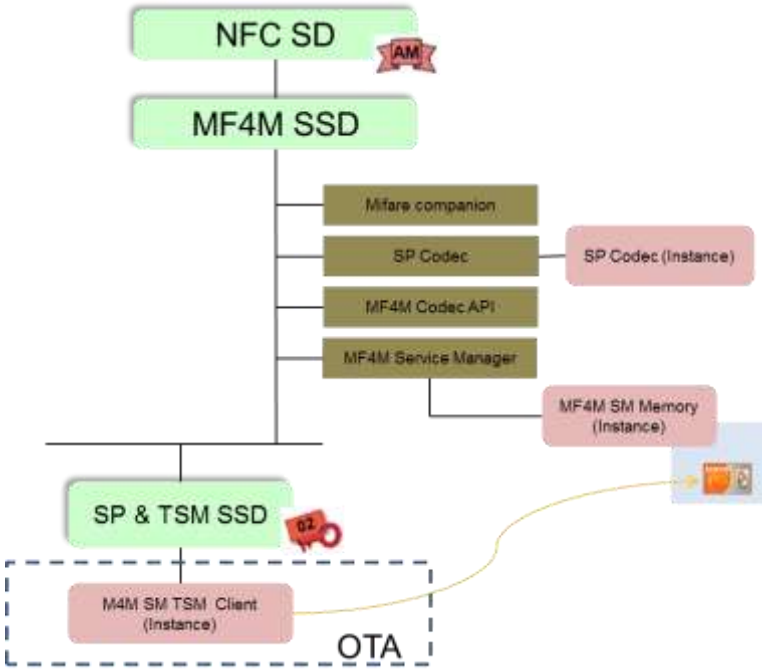


Figura 22 Perfil de Tarjeta SIM con Mifare4Mobile

4.3.2 Requisitos del usuario

El cliente final necesitará disponer de una tarjeta SIM y de un terminal móvil ambos NFC. Tener acceso a un navegador web para efectuar el registro y descargarse la aplicación Android.

4.4 Arquitectura

La Figura 3.5 muestra la arquitectura general del sistema. En ella podemos distinguir dos partes diferentes: el front-end, que es la parte del software que interactúa con el usuario o cliente final, y el back-end, que es la parte que procesa la entrada desde el front-end y que generalmente devuelve una respuesta.

En esta sección se enumerarán los módulos contenidos en cada una de las partes y en el capítulo 4 se describirá la funcionalidad de cada módulo con un mayor grado de detalle.

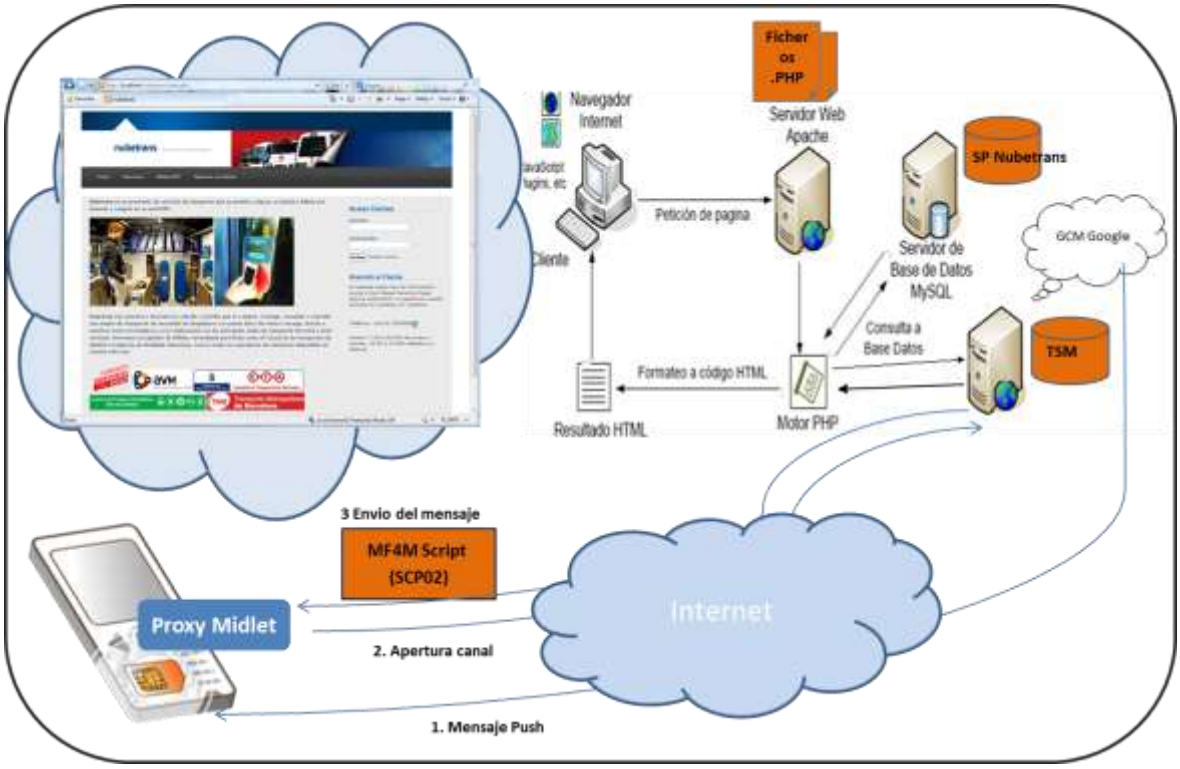


Figura 23 Arquitectura general del sistema Nubetrans

En la parte Back-end, el sistema cuenta con:

- **Servidor Web Apache HTTP**

  El servidor web se encuentra a la espera de que algún navegador le haga alguna petición HTTP, como por ejemplo, acceder a la página web de Nubetrans, y responde a la petición, enviando código HTML mediante una transferencia de datos en red. En nuestro caso hemos elegido Apache porque es el Servidor Web más utilizado, líder con el mayor número de instalaciones a nivel mundial muy por delante de otras soluciones como el IIS (Internet Information Server) de Microsoft. Apache es un proyecto de código abierto y uso gratuito, multiplataforma, muy robusto y que destaca por su seguridad y rendimiento.

- **Páginas PHP**

  Para generar páginas web dinámicas donde los contenidos pueden cambiar en base a los cambios que haya en una base de datos, de búsquedas o aportaciones de los usuarios, decidimos utilizar el lenguaje de programación PHP. El servidor web interpreta el código con un módulo de procesador de PHP que genera la página Web resultante.

- **Hojas de Estilo CSS "Madevo"**

  Las hojas de estilo dotan de presentación y aspecto a la página web. En Nubetrans para poder centrarnos en los aspectos técnicos de la solución se utilizará el estilo Madevo disponible en el foro phpBB http://www.phpbb3styles.net/

- **Base de Datos del Proveedor de Servicio MySQL**

  Utilizaremos una base de datos MySQL para almacenar la información de los clientes (credenciales, número de teléfono, servicios utilizados, balance etc) y de los servicios ofrecidos por Nubetrans (identificador de servicio, mapa MIFARE etc). En el siguiente capítulo describiremos el Modelo E-R (Entidad-Relación) diseñado.

- **Servidor de Aplicaciones (Java Spring framework sobre Tomcat 8)** [18]

  Para el servidor de aplicaciones o TSM, se ha decidido utilizar Spring como Framework (J2EE) corriendo sobre el servidor de aplicaciones Tomcat en su última versión, la 8.028. Apache Tomcat es la implementación de código abierto de las tecnologías Java Servlet, JavaServer Pages, Java Expression Language y Java WebSocket definidas y mantenidas por la comunidad Java, así mismo es uno de los servidores de aplicaciones de código abierto más utilizados en la industria.

- **Base de Datos del TSM JPA (Java Persistance API)**

  En el servidor de aplicaciones se ha hecho uso de las funcionalidades de persistencia de Java EJB 3.0 a través del uso de JPA (Java Persistance API). Este API nos permite a definir una base de datos relacional usando objetos de Java y hacerlos que persistan durante todo el ciclo de vida del servicio. Es un servicio ligero y a su vez completamente seguro e integrado en el Framework Spring.

En la parte Front-end.

- **Aplicación web**

  Son las paginas HTML resultantes del procesado del código PHP en el servidor web que se presentan al usuario. La aplicación web permite al usuario conocer los servicios ofrecidos por Nubetrans. Una vez registrado el usuario podrá gestionar de manera remota los títulos de transporte ofertados en su móvil NFC.

- **Aplicación y Servicio Android**

  En la parte de usuario o front-end, específicamente en el dispositivo móvil se ha creado un servicio de Android también denominado Android Proxy que se encargará de la comunicación con el TSM, pasando los comandos de este último hacia la tarjeta SIM para la gestión del ciclo de vida de los servicios M4Mobile.

Capítulo 5

Nubetrans

5.1 Introducción

En este capítulo se describen en detalle todas las partes implementadas que forman parte de sistema Nubetrans. Empezaremos describiendo la aplicación que implementa la lógica del TSM, ya que constituye el núcleo del sistema por encargarse de formatear los mensajes que gestionan el servicio MIFARE4Mobile en la tarjeta SIM. Seguiremos con la base de datos encargada de almacenar la información de los clientes y servicios de Nubetrans. Finalmente abarcaremos la aplicación web.

5.2 TSM (Trusted Service Manager)

La plataforma del TSM recibe las peticiones del Proveedor de Servicios Nubetrans y genera los comandos GlobalPlatform y MIFARE4Mobile junto con la seguridad SCP02 requerida para establecer una comunicación de canal seguro con la tarjeta SIM. Recordemos el flujo de comandos necesario para la gestión de contenido en una tarjeta GlobalPlatform definido en la sección 2.4.5 :

1. SELECCIÓN del Dominio de Seguridad
2. AUTENTICACIÓN
3. COMANDO/S (aplicando nivel de seguridad SCP02 escogido, véase 3.11)

Nuestro TSM soporta concretamente las operaciones de emisión, recarga y borrado de un Objeto de Servicio MIFARE4Mobile, expuestas en la Tabla 13, aplicando cifrado y MAC como nivel de seguridad.

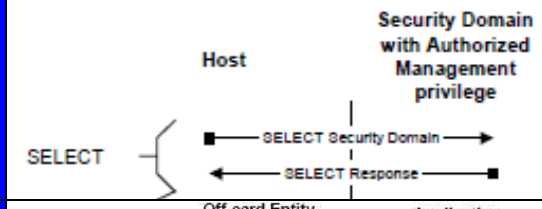
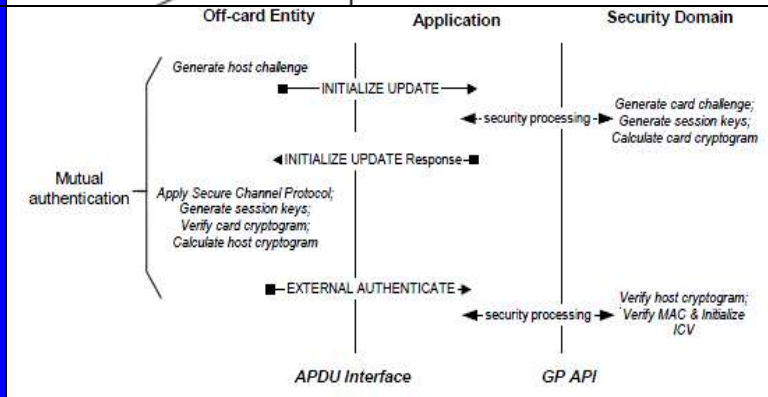
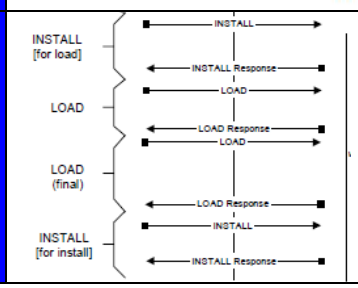
	EMISIÓN	RECARGA	BORRADO
Selección			
Autenticación			
Comandos (con seguridad SCP02)		UPDATE DATA (véase 3)	DELETE (véase 3)

Tabla 13 Flujo de comandos en el TSM de Nubetrans

A continuación listamos las principales clases definidas en el TSM clasificadas en tres grupos:

1. Clase controladora de las interfaces REST:
   - **TSMController**: Controlador donde se definen las operaciones REST disponibles en el TSM.

2. Clases de gestión de datos:
   - **M4mData**: Entidad JPA que albergará los datos de la sesión Mifare.
   - **MifareSectors**: Entidad JPA que almacena la información de los 16 sectores de Mifare
   - **M4mDataRepository**: Repositorio o Base de Datos de persistencia que extiende el repositorio CRUD para almacenar los datos de la sesión Mifare.

3. Clases de generación de comandos:
   - **GlobalPlatform**: Clase con los métodos de GlobalPlatform necesarios para formatear los comandos y calcular la seguridad SCP02
   - **GPCrypto:** Clase de apoyo a la clase de GlobalPlatform y que contiene las funciones criptográficas necesarias para la generación de comandos SCP02

5.2.1 Controlador de las interfaces REST

En esta sección se describen las operaciones disponibles en la clase **TSMController** para gestionar las peticiones recibidas desde la aplicación web y las del proxy en el móvil:

1. **Operación new – POST**: Tras recibir la petición POST de la aplicación web, esta operación crea una nueva sesión MIFARE4Mobile con los datos recibidos y notifica a GCM (Google Cloud Messaging) para que envíe un mensaje PUSH al número de teléfono (MSISDN) indicado en la petición junto con la acción a ejecutar. El móvil una vez recibido el mensaje PUSH abrirá el canal para el intercambio de APDUs.

```
@RequestMapping(value="/new", method=RequestMethod.POST)
   public @ResponseBody Response newRequest(@RequestParam(value="key",
required=true) long key, @RequestBody M4mData m4mData) throws Exception
{
```

Ejemplo de mensaje JSON enviado desde la interfaz web:


```
{
  "action": "delete",
  "label": "0102030405060708",
  "msisdn": "34658796561",
  "soid": "0601040182D33302",
  "dateBegin": "0102030405060708",
  "dateEnd": "0102030405060708",
  "offset": "0000",
  "parserAID": "A0000002984D344D56310200",
  "mifareKeys": "FFFFFFFFFFFFFFFFFFFFFFFF",
  "acc": "78778869",
  "sectors": {
    "sector1" : "11111111111111111111111111111111111111111111111111111111111111111",
    "sector2" : "11111111111111111111111111111111111111111111111111111111111111111[…]111",
    "sector3" : "11111111111111111111111111111111111111111111111111111111111111111[…]111",
    "sector4" : "11111111111111111111111111111111111111111111111111111111111111111[…]111",
        […]
    "sector16" : "11111111111111111111111111111111111111111111111111111111111111111[…]111"
}
```


2. **Operación /MSISDN – GET:** Esta operación se llamará desde el terminal móvil para obtener el identificador de sesión correspondiente al MSISDN indicado en la petición. El TSM enviará el número de sesión junto con el primer APDU a ejecutar que será un comando SELECT.

```
@RequestMapping(value="/{MSISDN}", method=RequestMethod.GET)
   public @ResponseBody GPSession
getSessionID(@RequestParam(value="key", required=true) long key,
@PathVariable String MSISDN) throws InternalServerException{
```

3. **Operaciones /select, /auth – POST:** Una vez recuperado el número de sesión el terminal móvil comenzará a solicitar los comandos al TSM para realizar la autenticación SCP02 utilizando estas operaciones.

```
@RequestMapping(value="/{MSISDN}/select",method=RequestMethod.POST)
    public @ResponseBody GPSession
startInstallSession(@RequestParam(value="key", required=true) long key,
@RequestBody GPSession session, @PathVariable String MSISDN) {

@RequestMapping(value="/{MSISDN}/auth",method=RequestMethod.POST)
    public @ResponseBody GPSession
getResponse(@RequestParam(value="key", required=true) long key,
@RequestBody GPSession session, @PathVariable String MSISDN) {
```

4. **Operaciones /install , /update and /delete – POST:** Una vez terminada de forma satisfactoria la autenticación SCP02 entre la tarjeta SIM y el servidor, el terminal móvil comenzará a solicitar los comandos correspondientes a la operación solicitada por el servidor web, instalación de un nuevo servicio Mifare, borrado de un servicio Mifare o actualización de los datos de un servicio Mifare.

```
@RequestMapping(value="/{MSISDN}/install/{step}",method=RequestMethod.POST)
    public @ResponseBody GPSession
getResponse(@RequestParam(value="key", required=true) long key,
@RequestBody GPSession session, @PathVariable String MSISDN,
@PathVariable int step) throws InternalServerException {

@RequestMapping(value="/{MSISDN}/update/{step}",method=RequestMethod.POST)
    public @ResponseBody GPSession
getResponseUpdate(@RequestParam(value="key", required=true) long key,
@RequestBody GPSession session, @PathVariable String MSISDN,
@PathVariable int step) throws InternalServerException {


@RequestMapping(value="/{MSISDN}/delete/{step}",method=RequestMethod.POST)
    public @ResponseBody GPSession
getResponseDelete(@RequestParam(value="key", required=true) long key,
@RequestBody GPSession session, @PathVariable String MSISDN,
@PathVariable int step) throws InternalServerException {
```

5.2.2 Clases de gestión de datos

En esta sección se describen las clases definidas en el TSM para almacenar los datos de sesión Mifare:

1. **Clase M4mData**: que define las variables que almacenarán los datos de sesión M4mData

```
import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.OneToOne;

@Entity
public class M4mData {

@Id
@GeneratedValue(strategy=GenerationType.AUTO)
```

```
private long id = 0;

private String Action = null;
private String mSISDN = null;
private String SOID = null;
private String Label = null;
private String DateBegin = null;
private String DateEnd = null;
private String Offset = null;
private String ParserAID = null;
private String MifareKeys = null;
private String ACC = null;

@OneToOne(cascade = CascadeType.PERSIST)
private MifareSectors Sectors;

public M4mData() {

}
//Setters y Getters
}
```

2. **Clase MifareSectors:** – Clase que almacena la información de los 16 sectores de Mifare

```
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

@Entity
public class MifareSectors {

	@Id
	@GeneratedValue(strategy=GenerationType.AUTO)
	private long ID;
	private String Sector1 = null;
	private String Sector2 = null;
	private String Sector3 = null;
	private String Sector4 = null;
	private String Sector5 = null;
	private String Sector6 = null;
	private String Sector7 = null;
	private String Sector8 = null;
	private String Sector9 = null;
	private String Sector10 = null;
	private String Sector11 = null;
	private String Sector12 = null;
	private String Sector13 = null;
	private String Sector14 = null;
	private String Sector15 = null;
	private String Sector16 = null;

	public MifareSectors() {

	}
```

5.2.3 Clases de generación de comandos

El TSM de Nubetrans podrá generar comandos para las 3 operaciones siguientes desde las clases **GlobalPLatform** y **GPCrypto**:

- **Emisión de tarjetas de transporte**. Esta operación consiste en la carga e instalación remota de un Objeto de Servicio MIFARE4Mobile con el mapeo de datos MIFARE enviado por la aplicación Web.

- **Recarga de tarjetas de transporte:** La operación de recarga consiste en enviar el comando UPDATE DATA para actualizar los datos propietarios del Objeto de Servicio MIFARE. Entre los datos propietarios se encuentra el balance disponible en la tarjeta de transporte.

- **Borrado de tarjetas de transporte**: La operación de borrado se lleva a cabo cuando el usuario decide darse de baja de uno de los servicios contratados. En este caso el TSM generará un comando DELETE para revocar un Objeto de Servicio MIFARE4Mobile emitido con anterioridad.

A continuación explicaremos con más detalle los comandos implementados en el TSM para cada una de las operaciones mencionadas.

5.2.3.1 Emisión de una tarjeta de transporte NFC

La aplicación web, tras la acción del usuario, enviará una petición de compra incluyendo los datos del servicio MIFARE seleccionado por el usuario (extraídos de la base de datos) junto con el número de teléfono del usuario.

En nuestra función de java **getInstallCommands** a continuación se puede observar cómo se crean los comandos de instalación de un servicio MIFARE4Mobile v1. Se recomienda realizar la escritura completa de la memoria Mifare e inicializar claves y condiciones de acceso, por ello las cabeceras y números de comandos son siempre fijos aunque se puede observar que los datos del servicio se obtienen de forma dinámica.

La función **getWrappedAPDU** se encarga, una vez formado el comando en claro, de aplicar la encriptación SCP02 correspondiente a la sesión y añadir los datos extras necesarios (MAC).

```
public String getInstallCommands(int commandN, M4mData m4mData) throws
GPException {

            if(commandN == 1) return getWrappedAPDU("80E602041708" +
m4mData.getSOID() + "00000AEF08C6020400C902000000");

            if(commandN == 2) return getWrappedAPDU("80E801003A8108" +
m4mData.getLabel()+ "8208" + m4mData.getSOID() + "8308" +
m4mData.getDateBegin() + "8408" +m4mData.getDateEnd()+"8502" +
m4mData.getOffset() + "8A0C" + m4mData.getParserAID());

            if(commandN == 3) return getWrappedAPDU("80E80001D287D003" +
m4mData.getMifareKeys() + "03" + m4mData.getMifareKeys() + "03" +
m4mData.getMifareKeys() + "03" + m4mData.getMifareKeys() + "03" +
m4mData.getMifareKeys() + "03" + m4mData.getMifareKeys() + "03" +
m4mData.getMifareKeys() + "03" + m4mData.getMifareKeys() + "03" +
m4mData.getMifareKeys() + "03" + m4mData.getMifareKeys() + "03" +
m4mData.getMifareKeys() + "03" + m4mData.getMifareKeys() + "03" +
m4mData.getMifareKeys() + "03" + m4mData.getMifareKeys());

            if(commandN == 4) return getWrappedAPDU("80E80002428840" +
m4mData.getACC() + m4mData.getACC() + m4mData.getACC() + m4mData.getACC() +
m4mData.getACC() + m4mData.getACC() + m4mData.getACC() + m4mData.getACC() +
m4mData.getACC() + m4mData.getACC() + m4mData.getACC() + m4mData.getACC() +
m4mData.getACC() + m4mData.getACC() + m4mData.getACC() + m4mData.getACC());

            if(commandN == 5) return getWrappedAPDU("80E80003B886820300 20" +
m4mData.getSectors().getSector1() + "30" + m4mData.getSectors().getSector2() +
"30" + m4mData.getSectors().getSector3() + "30" +
m4mData.getSectors().getSector4());

            if(commandN == 6) return getWrappedAPDU("80E80004C430" +
m4mData.getSectors().getSector5() + "30"+ m4mData.getSectors().getSector6() +
"30" + m4mData.getSectors().getSector7() + "30" +
m4mData.getSectors().getSector8());

            if(commandN == 7) return getWrappedAPDU("80E80005C430" +
m4mData.getSectors().getSector9() + "30"+ m4mData.getSectors().getSector10() +
"30" + m4mData.getSectors().getSector11() + "30" +
m4mData.getSectors().getSector12());

            if(commandN == 8) return getWrappedAPDU("80E88006C430" +
m4mData.getSectors().getSector13() + "30"+ m4mData.getSectors().getSector14() +
"30" + m4mData.getSectors().getSector15() + "30" +
m4mData.getSectors().getSector16());

            if(commandN == 9) return getWrappedAPDU("80E6040411000008" +
m4mData.getSOID() + "010002C90000");
            return null;
      }
```

A la hora de diseñar el script se han tenido en cuenta los siguientes requerimientos:

- Todos los elementos obligatorios del Objeto de Servicio MIFARE4Mobile están presentes en los comandos LOAD representados por su TAG y en orden secuencial. Un TAG sólo puede ocurrir una vez dentro del bloque de comandos LOAD (véase 3.12.2.2).
- El bit 8 del parámetro P1 de los comando LOAD es "0" cuando quedan más comandos LOAD que ejecutar y cambia a "1" en el último LOAD para indicar que no hay más comandos de acuerdo con el estándar de tarjeta GP 2.2.1 **1.**
- El bit 0 del parámetro P1 de los comando LOAD es "0" cuando se cargan "datos" y es "1" cuando se cargan metadatos MIFARE4Mobile en línea con la especificación MIFARE4Mobile v1.0.1, 3 .
- El parámetro P2 de los comandos LOAD, se incrementa secuencialmente más 1 en conformidad al estándar de tarjeta GP 2.2.1, **1.**
- Se personalizan las claves y las condiciones de acceso del todos los sectores "trailer" MIFARE Classic 1K aunque no se usen para garantizar el buen funcionamiento de la memoria MIFARE en la tarjeta SIM.
- Se personalizan los datos de todos los sectores MIFARE Classic 1K excluyendo el bloque 0 del sector 0 de uso restringido al emisor de tarjetas,

Ejemplo de script en claro de carga e inicialización de un Objeto de Servicio MIFARE4Mobile:

```
'Install for Load SOID (e.g. 0954174DC60E44CE)
80E6020417 08 {SOID} 00000A EF 08 C6020400C902003A00

'LOAD Metadata: MIFARE4Mobile TAGS
80E80100{Length Data}
 81 {Length LABEL} {LABEL} 82 08 {SOID} 83 08 {DateBegin} 84 08 {DateEnd} 85 02
{Offset} 8A {Length ParserID} {ParserID }

'LOAD Mifare Keys - 16 sectors (e.g. A0A1A2A3A4A5B0B1B2B3B4B5)
80E80001 D2
87D003{KEYS}03{KEYS}03{KEYS}03{KEYS}03{KEYS}03{KEYS}03{KEYS}03{KEYS}03{KEYS
}03{KEYS}03{KEYS}03{KEYS}03{KEYS}03{KEYS}03{KEYS}03{KEYS}

'LOAD Access Conditions - 16 sectors (e.g. FF078069 )
80E8000242
8840{ACC}{ACC}{ACC}{ACC}{ACC}{ACC}{ACC}{ACC}{ACC}{ACC}{ACC}{ACC}{ACC}{ACC}{AC
C}{ACC}
```

```
'LOAD Data 1: (Sector 0, Sector 1, Sector2, Sector3)
80E80003 B8
8682030020{SECTOR 0– 32 bytes }30{SECTOR1 }30{SECTOR2 }30{SECTOR3 – 48 bytes}
'LOAD Data 2: (Sector 4, Sector5, Sector6, Sector7)
80E80004 C4
30{SECTOR4 }30{SECTOR5 }30{SECTOR6}30{SECTOR7}
'LOAD Data 3: (Sector 8, Sector9, Sector10, Sector11)
80E80005 C4 30{SECTOR8 }30{SECTOR9 }30{SECTOR10 }30{SECTOR11 }
'LOAD Data 4: (Sector 12, Sector13, Sector14, Sector15)
80E88006 C4 30{SECTOR12 }30{SECTOR13}30{SECTOR14}30{SECTOR15 }

'Install for Install:
80E60404 11 000008{SOID} 010002C90000
```

Figura 24 Script en claro para cargar e instaciar un objeto MF4M

5.2.3.2 Recarga de una tarjeta de transporte NFC

La aplicación web, tras la acción del usuario, enviará una petición de recarga incluyendo el importe a recargar, el SOID de la tarjeta de transporte y el número de teléfono del usuario ambos extraídos de la base de datos de Nubetrans. En el TSM, la función **getUpdateCommand** se encarga de la construcción del mensaje para actualizar el balance en la tarjeta SIM:

```
public String getUpdateCommand(M4mDataRepository m4mDataRepo, String MSISDN)
throws GPException {

	String response = null;
	M4mData m4mData = m4mDataRepo.findByMSISDN(MSISDN);

	for(int i = 0 ; i < 16 ; i++) {
	//If sector exists in m4mData, then create APDU UPDATE DATA in clear
	if(m4mData.getSectors().getSector(i+1) != null) {
		//If sector is 0,APDU length is 32 bytes
		if(i == 0)
			response = "84EFFF2B2C08" + m4mData.getSOID() + "22" +
			HexUtils.encodeHexString(new byte[]{(byte)((i+1) & 0xFF)}) +
			"20" + m4mData.getSectors().getSector(i+1);
		else
			response = "84EFFF2B3C08" + m4mData.getSOID() + "32" +
			HexUtils.encodeHexString(new byte[]{(byte)((i+1) & 0xFF)}) +
			"30" + m4mData.getSectors().getSector(i+1);
		m4mDataRepo.delete(m4mData);
		m4mData.getSectors().setSector(i+1, null);
		m4mDataRepo.save(m4mData);
		break;
	}
	}
	return getWrappedAPDU(response);
}
```

Esta función leerá de la base de datos **m4mDataRepo** indexada por MSISDN, los sectores que se vayan a actualizar e irá componiendo el comando uno por uno, ya que MIFARE4Mobilr sólo permite la actualización de un sector por comando. Al igual que con la función de instalación, el comando final se enviará a la función **getWrappedAPDU** para calcular la encriptación SCP02 y más tarde enviarla a la tarjeta SIM.

5.2.3.3 Borrado de una tarjeta de transporte NFC

La aplicación web, tras la acción del usuario, enviará una petición de baja del servicio incluyendo el SOID de la tarjeta de transporte a borrar y el número de teléfono del usuario ambos extraídos de la base de datos del proveedor de servicios Nubetrans.

La función Java del TSM que se encarga de componer el mensaje de borrado es **getDeleteSOIDCommand**:

```
public String getDeleteSOIDCommand(M4mData m4mData) throws GPException {
            return getWrappedAPDU("80E400040A4F08" + m4mData.getSOID());
      }
```

5.2.3.4 Seguridad SCP02

Cualquier comando privilegiado que se quiera ejecutar en la tarjeta SIM requiere de la aplicación de la encriptación correspondiente, en nuestro caso, hemos decidido utilizar el modo SCP02 i=55 3.11 con mensaje cifrado y MAC. El TSM se encargará de aplicar esta seguridad a los mensajes, a la vez de generar el script completo de instalación incluyendo todos los comandos que se detallan a continuación. Las funciones Java del TSM encargadas de la creación de los comandos son las siguientes:

getSelect: Comando que selecciona el Service Manager que se encargará de crear el servicio Mifare en la SIM

```
public String getSelect(AID sdAID) throws GPException {
            CommandAPDU command;
            if (sdAID == null ) {
                  command = new CommandAPDU(ISO7816.CLA_ISO7816,
ISO7816.INS_SELECT, 0x04, 0x00, 256);
            } else {
                  command = new CommandAPDU(ISO7816.CLA_ISO7816,
ISO7816.INS_SELECT, 0x04, 0x00, sdAID.getBytes(), 256);
            }
            this.sdAID = sdAID;
            return HexUtils.encodeHexString(this.sdAID.getBytes());
      }
```

getInitAPDU– Esta función creará el comando INITIALIZE UPDATE e iniciará alguno de los valores necesarios en la configuración de SCP02 como puede ser el host_challenge. 3.11.3

```
public String getInitAPDU(GPKeySet staticKeys) {

	byte[] host_challenge;

	if (sdAID == null) {
		throw new IllegalStateException("!ISD No seleccionado!");
	}

	this.staticKeys = staticKeys;

	host_challenge = new byte[8];
	SecureRandom sr = new SecureRandom();
	sr.nextBytes(host_challenge);

	CommandAPDU initUpdate = new CommandAPDU(CLA_GP, INS_INITIALIZE_UPDATE,
staticKeys.getKeyVersion(), staticKeys.getKeyID(), host_challenge, 256);
		return HexUtils.encodeHexString(initUpdate.getBytes());
	}
```

getExtAuth – Aquí se inicializarán y generarán los elementos necesarios para el completo funcionamiento de SCP02. Esta función recibe como argumento la respuesta de la SIM al comando anterior pues hay ciertos valores de esta respuesta necesarios para la correcta inicialización de la encriptación SCP02. Una de las partes claves de esta función es la recepción "card challenge" y criptograma devuelto por la tarjeta como respuesta al comando INITIALIZE UPDATE.

```
public String getExtAuth(String responseAPDU, byte[] host_challenge)
			throws GPException {

		GPKeySet sessionKeys = null;

	-------------------------------------------------

	int offset = 0;
	diversification_data = Arrays.copyOfRange(update_response, 0, 10);
	offset += diversification_data.length;
	int keyVersion = update_response[offset] & 0xFF;
	offset++;
	scpMajorVersion = update_response[offset];
	offset++;

	byte card_challenge[] = Arrays.copyOfRange(update_response, offset,
offset + 8);
	offset += card_challenge.length;
	byte card_cryptogram[] = Arrays.copyOfRange(update_response, offset,
offset + 8);
	offset += card_cryptogram.length;
```

La siguiente parte de getExtAuth comprueba que la versión de las claves configurada en el servidor es la misma que la configurada en la tarjeta SIM:

```
.
            if ((staticKeys.getKeyVersion() > 0) && (keyVersion !=
staticKeys.getKeyVersion())) {
                  throw new GPException("la version de las claves no
concuerda: " + staticKeys.getKeyVersion() + " != " + keyVersion);
            }
```

Por último se muestra el final del proceso donde se crean las claves de sesión 3.11.2 derivando las claves estáticas y se crean los criptogramas para el mensaje EXTERNAL AUTHENTICATE.

```
byte [] seq = null;
            seq = Arrays.copyOfRange(update_response, 12, 14);
            sessionKeys = deriveSessionKeysSCP02(staticKeys, seq, false);

            byte[] my_card_cryptogram = null;
            byte[] cntx = GPUtils.concatenate(host_challenge, card_challenge);
            my_card_cryptogram =
GPCrypto.mac_3des_nulliv(sessionKeys.getKey(KeyType.ENC), cntx);

            if (!Arrays.equals(card_cryptogram, my_card_cryptogram)) {
                  printStrictWarning("Criptograma no valido!\nCard: " +
HexUtils.encodeHexString(card_cryptogram) + "\nHost: "+
HexUtils.encodeHexString(my_card_cryptogram));
            } else {
                  verbose("Criptograma verificado: " +
HexUtils.encodeHexString(my_card_cryptogram));
            }

            byte[] host_cryptogram = null;
            host_cryptogram =
GPCrypto.mac_3des_nulliv(sessionKeys.getKey(KeyType.ENC),
GPUtils.concatenate(card_challenge, host_challenge));
            wrapper = new SCP0102Wrapper(sessionKeys, scpVersion,
EnumSet.of(APDUMode.MAC), null, null);

            ------------------------------------

            CommandAPDU externalAuthenticate = new CommandAPDU(CLA_MAC,
ISO7816.INS_EXTERNAL_AUTHENTICATE_82, P1, 0, host_cryptogram);
            return
getWrappedAPDU(HexUtils.encodeHexString(externalAuthenticate.getBytes()));

      }
```

Una vez se reciba la respuesta de este comando y siendo satisfactoria (9000) el TSM estará listo para encriptar las APDUs de instalación/borrado o actualización del servicio Mifare 4 Mobile. La función **wrap** calcula la MAC y cifra el comando si es necesario como se muestra a continuación:

```
public CommandAPDU wrap(CommandAPDU command) throws CardException {

try {
       if (!mac && !enc) {return command;}
--------------------------------------------------------------------------------

       if (origLc > getBlockSize()) {
              throw new IllegalArgumentException("APDU too long for wrapping.");
       }

       if (mac) {
              if (icv == null) {
                     //Create ICV to 0
                     icv = new byte[8];
              } else if (icvEnc) {
                     Cipher c = null;
                     c = Cipher.getInstance(GPCrypto.DES_ECB_CIPHER);
                     c.init(Cipher.ENCRYPT_MODE,
                     sessionKeys.getKey(KeyType.MAC).getKey(Type.DES));
                     icv = c.doFinal(icv);
              }
--------------------------------------------------------------------------------
if (enc && (origLc > 0)) {

       t.write(GPCrypto.pad80(origData, 8));
       newLc += t.size() - origData.length;
       Cipher c = Cipher.getInstance(GPCrypto.DES3_CBC_CIPHER);
       c.init(Cipher.ENCRYPT_MODE, sessionKeys.getKeyFor(KeyType.ENC),
       GPCrypto.iv_null_des);
       newData = c.doFinal(t.toByteArray());
       t.reset();
       // CLA with security 84
       t.write(newCLA);
       t.write(origINS);
       t.write(origP1);
       t.write(origP2);
       if (newLc > 0) {
              t.write(newLc);
              t.write(newData);
       }
       if (mac) {t.write(icv);}
       if (le > 0) {t.write(le);}
       CommandAPDU wrapped = new CommandAPDU(t.toByteArray());

--------------------------------------------------------------------------------
                     }
              }
```

5.3 Base de datos

5.3.1 Modelo E/R

El modelo E/R o Entidad-Relación es una herramienta utilizada para el modelado de los datos en un sistema de información. En el diagrama se reflejan las entidades relevantes del sistema así como sus interrelaciones y propiedades. El objetivo del modelo entidad-relación es representar los objetos de la base de datos como entidades con atributos y que se vinculan entre ellas mediante relaciones.

Se considera entidad a cualquier tipo de objeto o concepto sobre el que se recoge información. Las entidades se representan gráficamente mediante rectángulos y su nombre, identificativo y unívoco, aparece en el interior.

Una relación es la correspondencia o asociación entre dos o más entidades. Cada relación tiene un nombre que describe su función. Las relaciones se representan gráficamente mediante rombos y su nombre identificativo aparece en el interior.

En la base de datos de Nubetrans se almacenan tanto usuarios como servicios.

Los usuarios son los clientes que se registran para poder gestionar remotamente (comprar, recargar o borrar) uno, varios o ningún servicio de transporte en su SIM NFC. Si un usuario se borra de la base de datos, no se borrarán los servicio/s que estaba gestionando.

Los servicios son tarjetas Mifare de diferentes operadores de transporte disponibles en la base de datos para ser compradas por usuarios. Cada servicio tiene entre 1 y N datos que no pueden ser borrados de la base de datos y que se utilizan para personalizar remotamente el servicio en la SIM NFC del usuario.

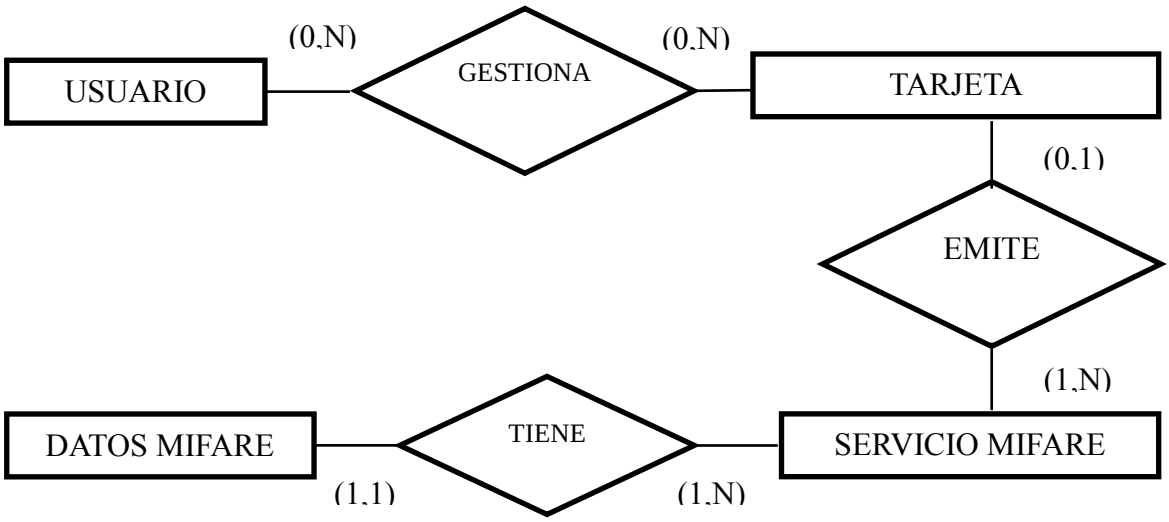


Figura 25 Modelo E/R

5.3.2 Modelo relacional

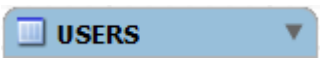
El modelo relacional para la gestión de una base de datos es un modelo de datos basado en la lógica de predicados y en la teoría de conjuntos. Su idea fundamental es el uso de las relaciones.

El modelo relacional fue propuesto por E.F. Codd en los laboratorios de IBM en California. Se trata de un modelo lógico [Irene Luque Ruiz - Ed. Ra-ma] que establece una estructura sobre los datos, aunque éstos posteriormente puedan ser almacenados de múltiples formas para aprovechar características físicas concretas de la máquina sobre la que se implante la base de datos realmente.

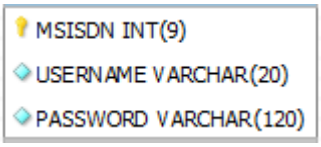
En este modelo todos los datos son almacenados en relaciones, y como cada relación es un conjunto de datos, el orden en que estos se almacenen no tiene relevancia. Esto tiene como ventaja la facilidad para entender y utilizar por un usuario no experto, la información puede ser recuperada o almacenada por medio de consultas que ofrecen una amplia flexibilidad y poder para administrar la información.

A continuación se van a definir las pautas seguidas para la representación del esquema que ayudarán al entendimiento del mismo:

- **Entidad**: Representa una "cosa" con existencia propia. En este caso las entidades son: Users, Service_Mifare, Service_Data y Cards. En la figura 4.2 son cada una de las tablas.



- **Atributos**: Son las propiedades que definen a una entidad. Por ejemplo, en el caso de Users sus atributos son: MSISDN, Username y Password. Algunos atributos específicos se denominan claves. Se llaman claves porque a partir de este atributo (o conjunto de atributos) podemos identificar inequívocamente a un individuo de esa entidad de cualquier otro (algo así como identificar objetos de una misma clase). En la figura 4.2 los atributos se listan en el interior de cada tabla



- **Relaciones**: es cómo interactúan entre sí las entidades. Hay diferentes tipos de relaciones teniendo en cuenta su cardinalidad. Los tipos de cardinalidades son los siguientes:

  - 

Uno a Uno (1…1) : Una entidad A se relaciona únicamente con una entidad B y viceversa. Por norma general una de las entidades pasará a ser atributo de la otra.
  - 

Uno a Varios (1..N): Una entidad A se relaciona con cero o varias entidades B, pero la entidad B sólo se relaciona con una única entidad de A. Un ejemplo de esta relación es la existente entre usuarios (USERS) y tarjetas (CARDS). Un usuario puede tener 0..N tarjetas de diferentes operadores de transporte pero una tarjeta sólo pertenecerá a un único usuario. Esta relación se traduce en el modelo físico en una FK (clave foránea) de la entidad A (en nuestro ejemplo usuario) en la entidad B (en este caso tarjeta).
  - 

Varios a Varios (N..M): Una entidad A se relaciona con cero o varias entidades B y viceversa.. Esta relación se traduce en la creación de una tabla intermedia donde se vincularan las claves de ambas entidades.

Una vez hemos explicado las partes del diagrama relacional, mostramos el modelo al completo:

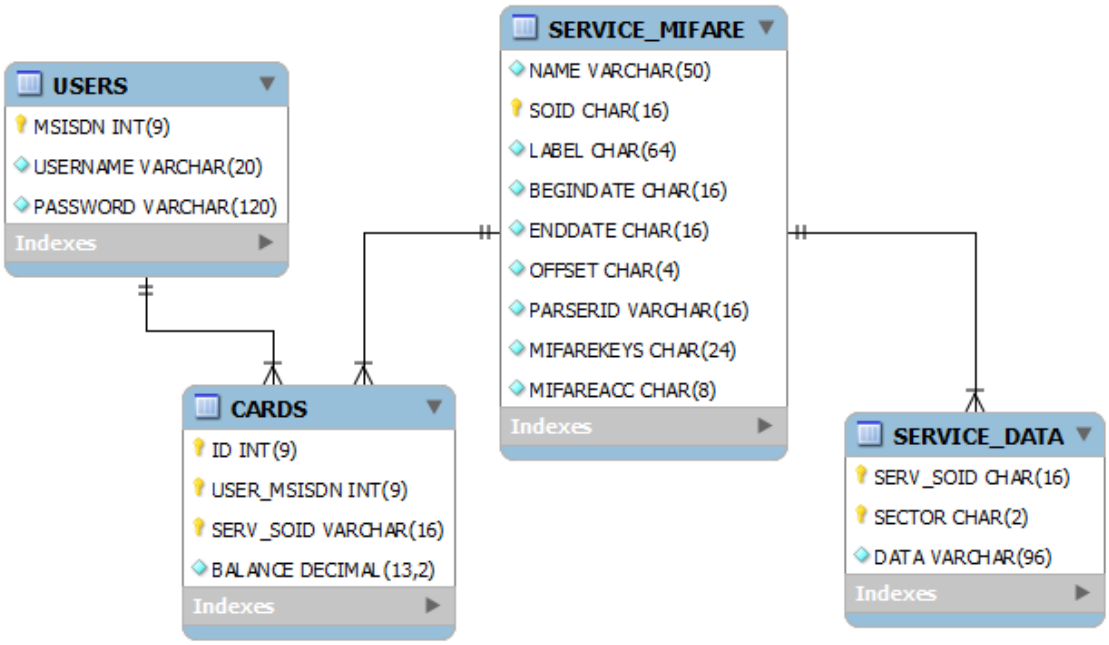


Figura 26 Modelo Relacional

Y a continuación se listan todas las tablas y se detallan los atributos en cada una de ellas:

Tabla 14 Base de datos Nubetrans, tabla SERVICE_MIFARE

SERVICE_MIFARE	Listado de los metadatos MF4M por servicio
NAME (UNIQUE)	Nombre alfanumérico del servicio
SOID (PK)	Identificador alfanumérico de servicio Mifare4Mobile
LABEL	Etiqueta Mifare alfanumérica asociada al servicio
BEGINDATE	Fecha de inicio (milisegundos desde Epoch) de servicio
ENDDATE	Fecha de fin (milisegundos desde Epoch) de servicio
OFFSET	Sector Mifare alfanumérico en el que se carga el servicio
PARSERID	Identificador alfanumérico del parser/códec Mifare que requiere el servicio en la SIM
MIFAREKEYS	Claves Mifare de servicio
MIFAREACC	Condiciones de acceso Mifare de servicio

Tabla 15 Base de datos Nubetrans, tabla SERVICE_DATA

SERVICE_DATA	Listado de los datos Mifare por servicio
SERV_SOID (PK, FK)	Identificador alfanumérico de servicio Mifare4Mobile
SECTOR (PK)	Sector Mifare alfanumérico
DATA	Datos Mifare de servicio

Tabla 16 Base de datos Nubetrans, tabla USERS

USERS	Listado de usuarios de Nubetrans
MSISDN (PK)	Número de teléfono como identificador numérico de usuario, de uso interno al sistema.
USERNAME (UNIQUE)	Alias utilizado por el usuario en el sistema Nubetrans (sensible a mayúsculas y minúsculas y sin espacios).
PASSWORD	Contraseña del usuario para autenticarse en Nubetrans (hash SHA-1).

Tabla 17 Base de datos Nubetrans, tabla CARDS

CARDS	Listado de los servicios comprados por los usuarios
ID (PK)	Identificador numérico de tarjeta
USER_MSISDN (PK, FK)	Identificador numérico de usuario
SERV_SOID (PK, FK)	Identificador alfanumérico de servicio Mifare4Mobile
BALANCE	Balance decimal

5.4 Aplicación Web

5.4.1 Diagrama de navegación

Diseñar el diagrama de navegación consiste en describir dinámicamente el orden y tipo de pantallas que se van produciendo durante la ejecución de la aplicación. Consiste en un autómata finito que describe la transición de ventanas para conocer de este modo las ventanas que preceden y a las que se pueden acceder desde cada una de las mismas.

En la Figura 27 se puede observar el diagrama de navegación de la aplicación realizada:

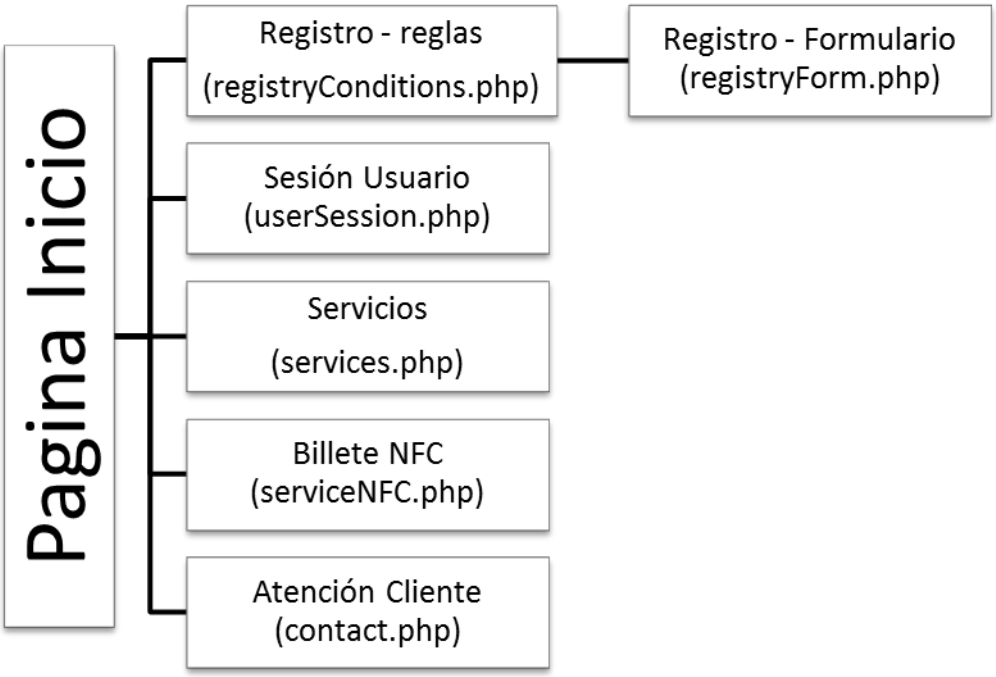


Figura 27 Diagrama de navegación de la aplicación web

Como se muestra en el diagrama de navegación, desde la página de Inicio se puede acceder a casi todas las ventanas de la aplicación. Comenzaremos hablando de la zona de "Acceso Clientes", situada a la derecha, por ser la que implementa la mayor funcionalidad.

Figura 28 Nubetrans. Acceso Clientes

Si el usuario quiere utilizar los servicios de Nubetrans, primero tendrá que registrarse. El link "Hazte Cliente" le llevará a una primera ventana (**registryConditions.php**) con los términos legales que tendrá que aprobar para acceder al formulario de registro **(registryForm.php**).

Figura 29 Nubetrans: Registro, términos legales

Figura 30 Nubetrans: Registro, formulario

En el formulario de registro se pedirá al usuario que se descargue la aplicación Android de Nubetrans en su móvil para habilitar la gestión remota. El potencial cliente tendrá que introducir su teléfono móvil, un nombre de usuario y una contraseña para crear su perfil en la base de datos de Nubetrans. Tras pulsar el botón "Enviar", se realizan las siguientes acciones:

1. Se valida el formulario. Se comprueba que no hay campos vacíos, que el formato de los datos es correcto y que el valor de la contraseña confirmada coincide con la contraseña elegida.
2. Se abre una conexión con la base de datos MySQL de Nubetrans
3. Se lanza una consulta para comprobar que el nombre de usuario o el número de teléfono no existan previamente
4. Se inserta un nuevo usuario

Una vez que el usuario ya está registrado, puede acceder a su perfil introduciendo su nombre de usuario y su contraseña desde la zona "Acceso Clientes". Al presionar el botón entrar, la aplicación web verifica que los credenciales son correctos y muestra la siguiente ventana, **userSession.php**, donde el cliente podrá gestionar sus tarjetas de transporte.

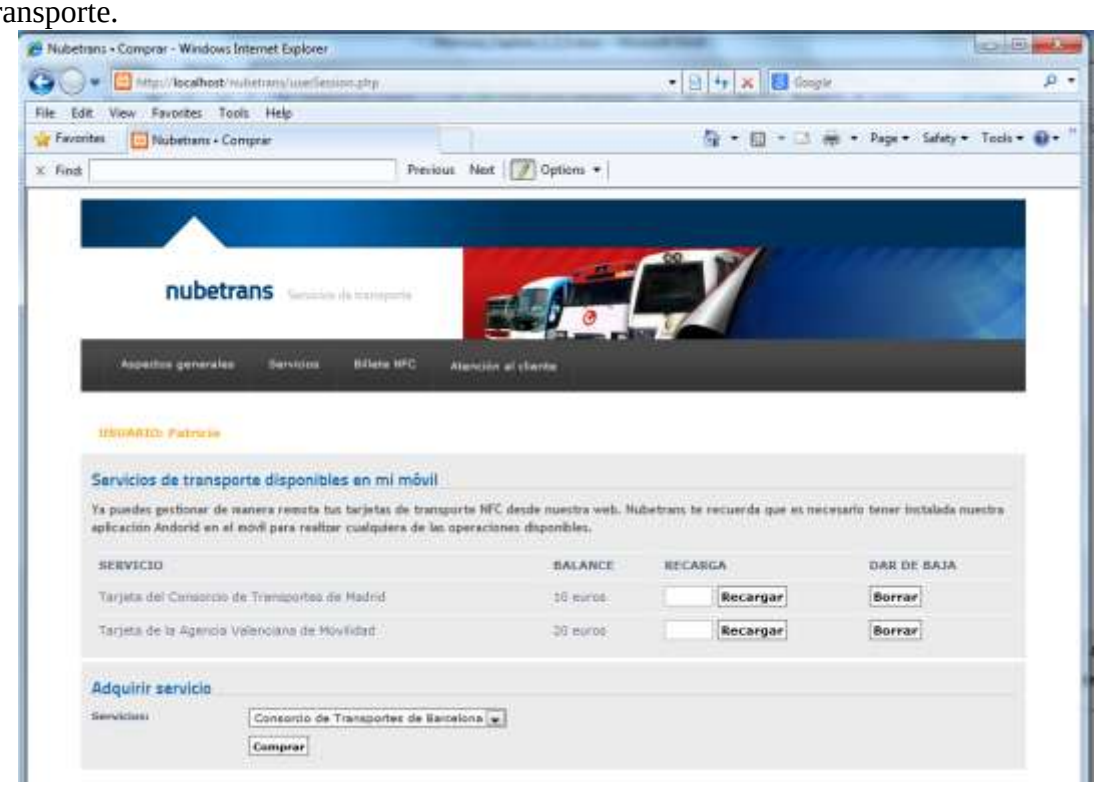


Figura 31 Nubetrans: Sesión de usuario

La opción “Adquirir servicio” ofrece la posibilidad al usuario de comprar la tarjeta de transporte NFC de cualquiera de los operadores disponibles en la lista desplegable. Una vez seleccionado el servicio, al pulsar el botón “comprar”, se siguen los siguientes pasos en la página “**userSession.php**”

1. Se abre conexión a la base de datos MySQL de Nubetrans
2. Se comprueba que el usuario no tenga dicho servicio ya contratado
3. Se extrae el número de teléfono del usuario y los datos del servicio Mifare seleccionado
4. Se crea la petición HTTP POST que enviará los datos al TSM para la creación del script MIFARE4Mobile con seguridad SCP02. En este paso se emplea JSON como formato de intercambio de datos, función “json_encode” en PHP.

```
<?php
…
$dataMifare = array(
"SECTOR1"=>$Sector1, "SECTOR2"=>$Sector2,
"SECTOR3"=> $Sector3,"SECTOR4"=>$Sector4, "SECTOR5"=>$Sector5,
"SECTOR6"=>$Sector6, "SECTOR7"=> $Sector7, "SECTOR8"=>$Sector8,
"SECTOR9"=>$Sector9,"SECTOR10"=>$Sector10, "SECTOR11"=>$Sector11,
"SECTOR12"=>$Sector12,"SECTOR13"=>$Sector13,"SECTOR14"=>$Sector14,
"SECTOR15"=> $Sector15, "SECTOR16"=>$Sector16);

$data = array("ACTION"=>$action, "MSISDN"=>$MSISDN, "SOID"=>$SOID,
"LABEL"=>$Label, "DATEBEGIN"=> $DateBegin, "DATEEND"=>$DateEnd,
"OFFSET"=>$Offset, "PARSERID"=>$ParserID, "KEYS"=>$MifareKeys, "ACC"=>$ACC,
"SECTORS"=>$dataMifare);

$data_string = json_encode($data);

$ch = curl_init('{ $TSM_URL}');
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "POST");
curl_setopt($ch, CURLOPT_POSTFIELDS, $data_string);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, array(
    'Content-Type: application/json',
    'Content-Length: ' . strlen($data_string))
);
```

5. Se recoge la respuesta y si la instalación remota del servicio en el móvil se ha hecho correctamente, se inserta los datos de la tarjeta de transporte nueva asociada al usuario en la tabla CARDS de la base de datos .

La misma lógica se aplicará en el caso de recargar o dar de baja una tarjeta de transporte previamente comprada e instalada, aunque con distintos datos JSON en la petición.

Desde la página de Inicio también se da acceso a la ventana "Servicios", **services.php,** donde se listan los operadores de transporte que pueden ser gestionados desde Nubetrans.

Figura 32 Nubetrans: Servicios

Capítulo 6

Distribución temporal y presupuesto

6.1 Introducción

Este proyecto surge dentro del marco laboral trabajando en el departamento de tarjetas SIM de Gemalto con la motivación personal de profundizar en los estándares Mifare4Mobile y GlobalPatform que definen la arquitectura para gestionar entornos móviles NFC dentro del ámbito de transporte.

Durante los meses de Marzo de 2015 a Octubre de 2015 se han atravesado diferentes etapas, las cuales serán expuestas en este capítulo. Se tratará de describir en profundidad las etapas del proyecto junto con una visión temporal y el presupuesto estimado del mismo.

6.2 Distribución temporal

Este proyecto se ha realizado en un tiempo aproximado de 8 meses. A continuación se describen brevemente las fases del proyecto junto al tiempo dedicado a cada una de ellas.

- **Análisis y planificación (90 días calendario)**

  El primer paso de este proyecto es la definición de criterios y objetivos así como la planificación de las diferentes acciones que surgen de los mismos. La documentación es clave para establecer una buena estimación de tiempos y recursos. En esta fase se adquieren conocimientos en los siguientes campos:

  - Formato de memoria MIFARE
  - Arquitectura e implementación Mifare4Mobile v1.01
  - Especificación de Tarjeta Global Platform 2.2.1 y protocolos de Seguridad
  - Requerimientos para crear tarjetas SIM NFC de Gemalto
  - Tecnologías para implementar la aplicación web del Proveedor de Servicio
  - Tecnologías para implementar la aplicación servidor, TSM en funciones.

  As su vez se diseña la base de datos de usuarios y servicios para la gestión de tarjetas de transporte Mifare.

- **Implementación (150 días calendario)**

  En esta fase además de la programación del código como tal, se realizó la búsqueda de la información necesaria para la implementación de las aplicaciones y la solución de problemas específicos que iban surgiendo. Se comenzó con la aplicación servidor por ser el elemento que gestiona la parte principal del proyecto, es decir, el formateo del script Mifare4Mobile y la seguridad, utilizando datos fijos. Se continuó con la aplicación web y la base de datos que proporcionaban al usuario un interfaz desde donde lanzar sus peticiones de compra o recarga. A su vez se actualizó la aplicación servidor para gestionar datos Mifare dinámicos de acuerdo a las peticiones de la aplicación web. Y finalmente se implementó una sencilla aplicación Android para recibir el script Mifare4Mobile en el móvil y pasárselo a la tarjeta SIM.

- **Pruebas de carga en tarjeta (30 días calendario)**

  Cuando se tuvo el prototipo de la aplicación servidor funcionando se hicieron pruebas de carga del script Mifare4Mobile generado por la aplicación en una tarjeta SIM de Gemalto con soporte NFC 2.1. Para realizar la carga, se insertó la tarjeta SIM en un lector con contactos conectado al PC y se ejecutó el script desde una herramienta PC de Gemalto. Una vez ejecutado el script, se hicieron pruebas

insertando dicha tarjeta SIM en un teléfono móvil real y leyendo su mapa de memoria Mifare, esta vez con un lector de tarjetas sin contacto activado en la herramienta MifareDemo 3.0. Las pruebas se repitieron hasta conseguir un script correcto con la aplicación servidor, en cuyo caso se pudo ver que el mapa de la memoria Mifare se había actualizado correctamente.

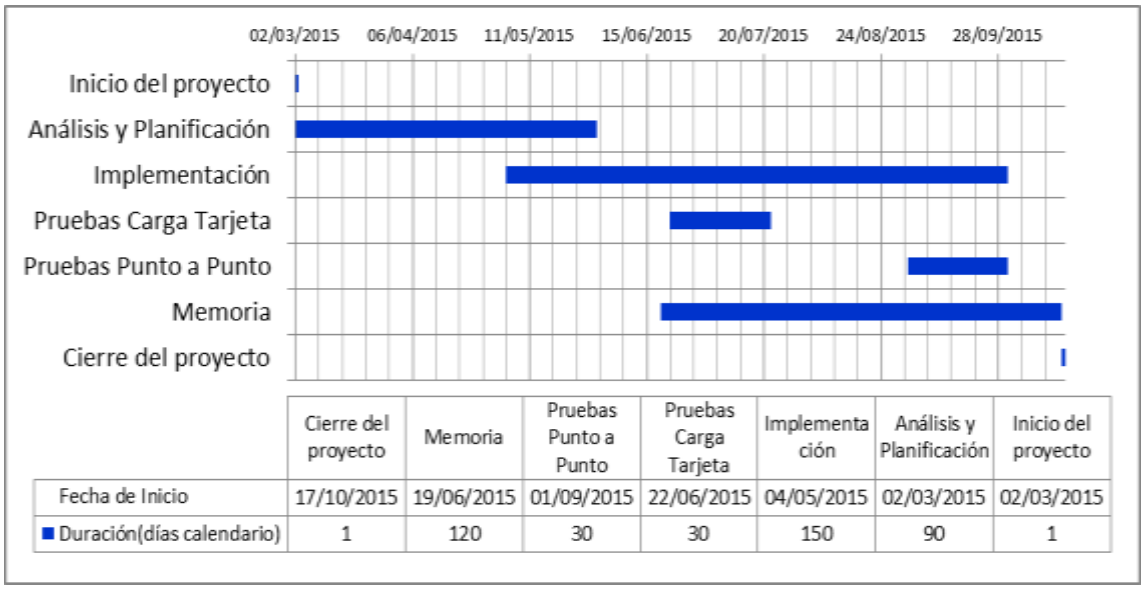
- **Pruebas Punto a Punto (30 días calendario)**

Una vez se tuvo todas las aplicaciones del sistema, se hicieron pruebas insertando la tarjeta SIM en el terminal LG y lanzando peticiones de gestión desde la aplicación web. El correcto funcionamiento se validó verificando el mapa de memoria Mifare resultante de la operación de la misma manera que en la pruebas de carga del script en tarjeta.

- **Realización de la memoria (120 días calendario)**

La realización de la memoria se lleva a cabo en un periodo aproximado de 4 meses teniendo en cuenta que en este periodo se ha simultaneado con otras tareas de implementación y pruebas en tarjeta.

En la Figura 5.1 se muestra el correspondiente diagrama de Gantt con las fases descritas anteriormente sobre una escala de tiempos de unos 8 meses, con una unidad mínima de división sobre el eje horizontal de 7 días calendario.



	Cierre del proyecto	Memoria	Pruebas Punto a Punto	Pruebas Carga Tarjeta	Implementación	Análisis y Planificación	Inicio del proyecto
Fecha de Inicio	17/10/2015	19/06/2015	01/09/2015	22/06/2015	04/05/2015	02/03/2015	02/03/2015
Duración(días calendario)	1	120	30	30	150	90	1

Figura 33 Diagrama de Gantt

6.3 Presupuesto del proyecto

En este apartado se presenta el presupuesto del proyecto, estableciendo tanto el coste material como el del trabajo de las personas que han participado en su desarrollo.

6.3.1 Costes de personal

Los costes de personal incluyen los honorarios del Ingeniero Técnico de Telecomunicación en Telemática encargado del desarrollo del proyecto. La duración de este proyecto ha sido de aproximadamente 8 meses, unas 33 semanas. Suponiendo 5 días laborables a la semana, se obtiene un total de 165 días laborables. Con una jornada laboral de cuatro horas diarias, la realización del proyecto ha requerido de 660 horas.

En la actualidad los honorarios profesionales se encuentran liberalizados. Según el baremo orientativo del Colegio Oficial de Ingenieros Técnicos de Telecomunicación (COITT)[24] antes de la liberación, los honorarios de un Ingeniero Técnico de Telecomunicación en Telemática son de 45 euros la hora. Por tanto, el coste total asciende a 29.700 euros la Tabla 18 recoge este resultado.

Concepto	Horas	Honorarios	Importe
Ingeniero Técnico Telecomunicación Telemática	660	45 €/hora	29700 €

Tabla 18 Costes de personal

6.3.2 Costes de material

Los materiales empleados durante la realización del proyecto han sido los siguientes (véase Tabla 19):

- Un ordenador personal portátil DELL Latitude E6440 con sistema operativo Windows 7, valorado aproximadamente en 1.099 €. Si consideramos que el portátil tiene una vida de 3 años y que lo hemos utilizado 8 meses para la realización del proyecto, el coste proporcional de amortización sería de 240 €.
- Un teléfono móvil LG Optimus L7 II, cuyo precio de acuerdo con la información extraída de la tienda online de LG (véase Figura 21 LG L7 II) es de 199 €. Puesto que se ha comprado el teléfono exclusivamente para el proyecto, aplicamos el coste total.

- El software utilizado es de distribución libre, tanto XAMPP v5.5.24/5.6.8 como ECLIPSE 4.5 por lo que no se incluye en los costes de material.
- Un Lector de tarjetas PC Prox DU Gemalto 48.95 euros y tarjetas inteligentes NFC 2.1 de Gemalto 1.5 euro la unidad
- Conexión a Internet durante la realización del proyecto. Necesaria para conseguir documentación. Está valorada aproximadamente en 42 euros al mes[28], que multiplicado por los meses de trabajo, supone un coste de 336 euros

Concepto	Unidades	Precio Unitario	Importe
Ordenador personal	1	240€	240€
Teléfono móvil LG	1	199€	199€
Lector de tarjetas Prox DU	1	48.95€	48.95€
Tarjetas inteligentes Gemalto	10	1.5€	15€
Conexión a Internet	1	336€	336€
TOTAL			838.95€

Tabla 19 Costes de material

6.3.3 Presupuesto total

El presupuesto final para la realización de este proyecto está formado por los costes de material y de personal presentados anteriormente. Como se observa en la Tabla 20, el presupuesto total de este proyecto asciende a la cantidad de 30538.95 euros.

Concepto	Importe
Costes de personal	29700 €
Costes de material	838.95 €
TOTAL	30538.95 €

Tabla 20 Coste Total

Capítulo 7

Conclusiones y trabajos futuros

7.1 Conclusiones

En este proyecto se ha evaluado la viabilidad y complejidad de implementar una solución basada en los estándares GlobalPlatform y MIFARE4Mobile v 1.01 que permita gestionar aplicaciones MIFARE en una tarjeta SIM NFC insertada en un móvil como otro medio de acceso a la red de transporte. Durante 2014 hemos visto los primeros lanzamientos de pago con el móvil en el transporte público en España, primero Valencia y después Logroño, una tendencia a la que se están uniendo cada vez más operadores de transporte. No hay duda de las ventajas de usar el móvil pero sí en qué implementación es más conveniente para el operador de transporte al evaluar el coste, interoperabilidad, complejidad y dependencias de cada una de las opciones disponibles. El uso de la tarjeta

SIM como medio para gestionar una tarjeta MIFARE hace indispensable acordar la arquitectura con los diferentes operadores móviles y puede ser que no todos ellos soporten una solución estándar como MIFARE4Mobile. Es por esto que considero apropiado profundizar en dicha tecnología y entender cuáles son las restricciones de algunos operadores móviles para optar por soluciones propietarias que fragmentan el mercado y añaden complejidad al ecosistema final. A continuación se describen tanto los resultados obtenidos como las limitaciones encontradas y las posibles causas.

Se ha probado que es posible implementar un TSM basado en los comandos y la seguridad definida en la especificación MIFARE4Mobile v1.0.1 de libre distribución compatible con una aplicación Java Card de un proveedor de tarjetas concreto, en este caso Gemalto, que implementa el Service Manager de MIFARE4Mobile v1.0.1. Se ha probado que es posible crear, actualizar y borrar de manera remota Objetos de Servicio MIFARE4Mobile v1.01 que contienen el mapa de memoria MIFARE de un servicio concreto en la tarjeta SIM. Se ha probado que utilizando la estructura TLV definida para el Objeto de Servicio en MIFARE4Mobile v1.01 es posible configurar un mapa de memoria MIFARE Classic 1K en la emulación de tarjeta MIFARE dentro de la tarjeta SIM. Se ha probado que desde una aplicación web es posible seleccionar de manera dinámica el servicio MIFARE que se desea descargar y que es posible tener varios servicios sobre la misma tarjeta SIM NFC.

Por otro lado nos hemos encontrado con un elemento cuya definición queda fuera de la especificación MIFARE4Mobile v1.0.1 y que juega un papel clave a la hora de actualizar o consular el mapa MIFARE de los Objetos de Servicio ya creados en la tarjeta SIM: el códec. En el caso concreto de Gemalto, hemos trabajado con el códec proporcionado que interpreta los comandos “UPDATE DATA” y “GET DATA” y que se comunica con el Sevice Manager a través del API que Gemalto pone a disposición en la tarjeta SIM. Sin embargo, dicho códec no sería compatible con la tarjeta SIM de otro proveedor, siendo necesaria su adaptación. En un ecosistema real con tarjetas SIM de diferentes proveedores, significaría tener un códec por fabricante de tarjeta para cada servicio, que por un lado mantuviera el mismo interfaz con el TSM y/o el wallet pero que implementara el API de cada proveedor para la comunicación con la tarjeta.

Otra limitación más evidente, es que MIFARE4Mobile v1.0.1 solo soporta MIFARE Classic 1K pero no MIFARE Classic 4K o DESFire, por lo que habría que descartar operadores de transporte como Madrid cuya infraestructura es completamente DESFire.

Con respecto a la seguridad, MIFARE4Mobile v1.0.1 solo soporta SCP01 y SCP02 como protocolos de comunicación segura. Para el envío del mensaje se puede utilizar el máximo nivel de seguridad que implica cifrado y MAC empleando algoritmos 3DES, pero la respuesta no puede ir cifrada, sólo firmada de acuerdo con lo definido en GlobalPlatform. Si pensamos en operaciones como GET DATA, nos damos cuenta que los datos MIFARE viajan en claro y que podrían ser interceptados por alguien que quisiera duplicar un mapa MIFARE para acceder a un servicio de transporte público.

También nos llama la atención la gestión del UID en MIFARE4Mobile v 1.01. MIFARE4Mobile v 1.01 define el concepto de "aplicación MIFARE" como una colección de datos que se puede cargar en la memoria física MIFARE del elemento seguro, pero no contempla la necesidad de tener diferentes UID para cada Objeto de Servicio. Digamos que podemos tener una única emulación de memoria MIFARE en la tarjeta SIM y que copiamos los datos de servicio cada vez que se cambia de un servicio a otro pero manteniendo el UID ya que es la misma tarjeta MIFARE. Si el usuario decide tener más de un servicio de trasporte, digamos Madrid y Valencia al mismo tiempo, ambos tendrían que compartir UID sin tener en cuenta los requerimientos de cada infraestructura de transporte. A día de hoy el UID no es un secreto, ya que está en claro en el mapa MIFARE pero se utiliza en la fase de autenticación por lo que algunos operadores de transportes son reacios a compartirlo.

Finalmente comentar que para comercializar el prototipo, habría que soportar la creación de dominios de seguridad remota, implementar los diferentes modos de funcionamiento GlobalPlatform y utilizar dispositivos HSM (Hardware Security Module) para la generación y almacenaje de las claves criptográficas SCP02. Hay que tener en cuenta que las claves SCP02 de los Proveedores de Servicio normalmente no vienen precargadas en las tarjetas SIM y que hay que poder inyectarlas de manera remota en caso de nuevos acuerdos comerciales a lo largo de la vida de la tarjeta. Tampoco hay que olvidar que las relaciones de confianza entre los Proveedores de Servicio y los Operadores de Telefonía Móvil pueden requerir el consentimiento de este último antes de ejecutar ninguna operación en la tarjeta SIM, para lo que el TSM tendría que soportar el modelo delegado de GlobalPlatform.

7.2 Líneas de trabajo futuras

Tras el estudio de MIFARE4Mobile v 1.01 y GlobalPlatform y pensando en proyectos futuros que complementen el sistema Nubetrans implementado, se proponen las siguientes líneas de trabajo futuro:

- **Implementar autenticación entre el Proveedor de Servicios y el TSM.** En la implementación actual no se utiliza ningún tipo de protocolo d autenticación entre el proveedor de servicio y el TSM, simplemente en las peticiones se indica un parámetro "*key*" que junto con un valor únicamente conocido por el cliente y el servidor permitirá la ejecución de las operaciones en este último. Como mejora futura se propone la utilización de un sistema de autenticación basado en tokens de acceso como Oauth2 u OpenID, de esta forma, el TSM mediante la emisión de

tokens o credenciales de acceso podrá gestionar y revocar si es necesario el permiso a los sistemas que intenten acceder a los recursos de éste.

- **Migración a MIFARE4Mobile v2.1.1.** La mayoría de las limitaciones de MIFARE4Mobile v1.01 descritas en el apartado anterior se podrán solucionar migrando a MIFARE4Mobile v2.1.1. La especificación de la versión 2.1.1 no estuvo lista hasta Abril de 2015, un poco tarde para considerarla en este proyecto. Entre las mejoras destacan el soporte de MIFARE Classic 4K y DESFire, la sustitución del concepto "Objeto de Servicio" por el de "Tarjeta Virtual" (VC – Virtual Card) y el soporte del algoritmo de seguridad SCP03 que utiliza AES y permite cifrar las respuestas a los comandos. Cada tarjeta virtual tiene su emulación de tarjeta MIFARE en la tarjeta SIM y por tanto su UID propio.

- **Implementar otros canales de gestión remota con el móvil**. Al no disponer de una subscripción de red con un Operador Móvil y por simplicidad, se decidió utilizar HTTPS y el servicio GCM (Google Cloud Messaging) para la enviar los comandos desde el TSM al móvil. En un ecosistema real, nuestro TSM tendría que soportar otros canales de gestión remota como son SMS (y el consecuente protocolo de seguridad SCP80) y BIP CAT/TP.

- **Pasarela de pagos**. Nuestra aplicación web acepta peticiones de compra y recarga sin llevar a cabo una transferencia real a través de una pasarela de pago electrónico. Se propone integrarlo en una posible fase comercial.

- **Implementar toda la lógica de servicio en la aplicación móvil**: En la versión actual sólo se contempla lanzar las operaciones de compra, recarga y borrado desde una página web pero sería más amigable para el usuario final poder gestionar todo desde una aplicación en el móvil ya que es donde almacenamos las tarjetas de transporte y contamos con conexión a la red para operaciones remotas.

- **Implementar el Modo delegado de Global Platform en el TSM**: En nuestro proyecto hemos partido de un perfil de tarjeta SIM concreto con un dominio de seguridad en modo GlobalPlatform autorizado que permite al Proveedor de Servicios la gestión de contenido dentro de su dominio sin necesidad de consentimiento por parte del Operador Móvil. El modo autorizado simplifica la implementación del TSM ya que no es necesario gestionar tokens con una tercera plataforma cada vez que se generan los comandos que se envían al móvil, pero no suele ser el elegido por los Operadores Móviles que sienten que pierden el control sobre una parte de su tarjeta.

- **Soportar la creación de dominios de seguridad remota y la inyección de claves en el TSM.** Nuestro perfil de tarjeta SIM cuenta con una estructura de dominios de seguridad precargada para el Proveedor de Servicios Nubetrans pero no suele ser el caso en un despliegue real, donde se necesita flexibilidad para poder crear la estructura de manera remota al momento de firmar los acuerdos comerciales. En tal caso nuestro TSM tendría gestionar certificados con un Autoridad de Control (CA) que permitan inyectar de manera segura las claves SCP02 vinculadas al nuevo Proveedor de Servicios en cada tarjeta SIM.

Glosario

AID	*Application Identifier*
APDU	*Application Protocol Data Unit*
API	*Application Programming Interface*
BIP	*Bearer Independent Protocol*
CLA	*CLAss, byte de clase de una APDU*
CSS	*Cascading Style Sheets*
ETSI	*European Telecommunications Standards Institute*
INS	*INStruction, byte de instrucción de una APDU*
IP	*Internet Protocol*
ISO	*International Organization for Standarization*
HTML	*HyperText Markup Language*
HTTP	*HyperText Transfer Protocol*
HTTPS	*HyperText Transfer Protocol Secure*
Lc	*Command Length, parámetro de una APDU*
Le	*Length Expected, parámetro de una APDU*
MF4M	*MiFare4Mobile*
NDEF	*NFC Data Exchange Format*
NFC	*Near Field Communication*
OTA	*Over The Air*
P1	*Primer parámetro en una APDU*
P2	*Segundo parámetro en una APDU*
PHP	*Hypertext Pre-Processor*
RFID	*Radio Frecuency Identification*
SIM	*Subscriber Identity Module*
SMS	*Short Message Service*
SQL	*Structured Query Language*
TCP/IP	*Transmission-Control-Protocol/Internet Protocol*
TISC	*Tarjeta Inteligente Sin Contacto*
3GPP	*3rd Generation Partnership Project*
UID	*Unique IDentifier*
URL	*Uniform Resource Locator*
USIM	*Universal Subscriber Identity Module*

Referencias

1. GlobalPlatform Card Specification 2.2.1. Documento disponible en la URL: https://www.globalplatform.org/specificationscard.asp (Enero 2011).

2. "GlobalPlatform's Proposition for NFC Mobile: Secure Element Management and Messaging" White paper (Abril 2009). Documento disponible en la URL: https://www.globalplatform.org/mediawhiteguides.asp (Abril 2009)

3. MIFARE4Mobile Interface Specification V1 01.pdf. Documento disponible en la URL: http://mifare4mobile.org/downloads/specifications_m4m/specifications-v211/ (Septiembre 2009).

4. MIFARE4Mobile_Whitepaper_V1.01.pdf (Enero 2012). Documento disponible en la URL: http://mifare4mobile.org/downloads/white/.

5. NFC Forum. Especificaciones disponibles en la siguiente URL: http://www.nfc-forum.org/specs/spec_license. Marzo 2009.

6. NFC Forum: "NFC in Public Transport" (1ª edición ISBN – 10: 84-616-4714-9 ISBN – 13: 978-84-616-4714-9 - Enero 2011).Documento disponible en la URL: http://nfc-forum.org/.

7. NFC Data Exchange Format (NDEF). Especicaciónn disponible en http://www.nfc-forum.org/specs/spec_list. Enero 2010.

8. NFC Forum: tabla "Comparing NFC to other close range communication technologies" traducida de inglés a español. URL: http://nfc-forum.org/.

9. ISO/IEC 14443: Identification cards. Contactless integrated circuit cards. Proximity cards. 2001, URL: www.iso.ch.Diciembre 2009.

10. ISO/IEC 7816-2:2007, URL: www.iso.ch. Figura extraida de la búsqueda "Tarjeta Inteligente" en http://es.wikipedia.org.

11. NXP, AN1304 NFC Type MIFARE Classic Tag Operation (Rev. 1.3 - 2 October 2012). Documento disponible en la URL http://www.nxp.com/.

12. Tarjeta Mifare, A3M proveedor de tarjetas plásticas, http://www.a3m.eu/es/tarjetas-plasticas/tarjetas-de-proximidad-rfid/tarjeta-mifare.html

13. Libro Blanco sobre la aplicación de la tecnología NFC en el transporte público, (Septiembre 2013). Disponible en la URL: http://www.fomento.gob.es/

14. Gemalto, "m-NFC Technology and products Description.". www.gemalto.com.

15. Gemalto, Lector Dual Prox DU para tarjetas in teligentes disponible en la URL http://www.gemalto.com/Products/prox_DU_SU/index.html

16. Gemalto. Guía de usuario para instalar el applet Java Card de MIFARE4Mobile v1.01 en una tarjeta SIM. Documento privado.

17. HTTP 1.1: Hypertext Transfer Protocol. HTTP/1.1. Disponible en la URL: http://www.w3.org/Protocols/rfc2616/rfc2616.html. Marzo 2010.

18. Framework Spring, http://spring.io/

19. GCM (Google Cloud Messaging) https://developers.google.com/cloud-messaging/

20. Tutoriales de PHP. http://www.programacion.com/php/tutorial/php/

21. Tutoriales SQL. www.mysql.com/dev.mysql.com/doc/refman/5.0/es/index.html.

22. WorldWideWebConsortium (W3C)- Oficina Española. 2008. http://www.w3c.es.

23. XAMPP Apache Friends. Sitio web acerca de la instalación y uso del paquete. http://www.apachefriends.org/es/xampp.html

24. Estándares NFC. http://www.nfc-world.com/en/about/03.html

25. Arquitectura de un móvil NFC. http://www.fqingenieria.com/es/conocimiento/tecnologia-nfc-modalidades-operativas-y-aspectos-tecnicos-47.

26. Proyecto Fin de Carrera "Desarrollo de un prototipo basado en las tecnologías NFC y SCWS para tarjetas inteligentes (U)SIM". Patricia de Noriega Vivas. Abril 2010.

27. Modos de funcionamiento NFC, http://www.myti.it/blog/2013/11/5/nfc-operating-modes

28. ETSI TS 102 226. Smart Cards; Remote APDU structure for UICC based applications (Release 8) http://www.etsi.org/.

.

Leganés a 29 de Octubre de 2015

El ingeniero proyectista

Fdo. Patricia Mozas Vozmediano